Trait-based modeling of larval dispersal in the Gulf of Maine

by

Benjamin Thomas Jones

B.S., Mathematics & Environmental Biology
Tulane University (2012)

Submitted to the Department of Biology
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computational Oceanography

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

and

WOODS HOLE OCEANOGRAPHIC INSTITUTION

September 2017

Author .....
Joint Program in Oceanography/Applied Ocean Science & Engineering
12 July 2017

Certified by .....
Rubao Ji
Associate Scientist with Tenure
Thesis Supervisor

Accepted by .....
Ann Tarrant
Chairman, Joint Committee for Biological Oceanography

Trait-based modeling of larval dispersal in the Gulf of Maine

by

Benjamin Thomas Jones

Submitted to the Department of Biology
on 12 July 2017, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computational Oceanography

Abstract

Population connectivity is a fundamental process that governs the spatial and temporal dynamics of marine ecosystems. For many marine species, population connectivity is driven by dispersal during a planktonic larval phase. The ability to obtain accurate, affordable, and meaningful estimates of larval dispersal patterns is therefore a key aspect of understanding marine ecosystems. Although field observations provide insight into dispersal processes, they do not provide a comprehensive assessment. Individual-based models (IBMs) that couple ocean circulation and particle-tracking models provide a unique ability to examine larval dispersal patterns with high spatial and temporal resolution. Obtaining accurate results with IBMs requires simulating a sufficient number of particles, and the sequential Bayesian procedure presented in chapter 2 identifies when the number of particles is adequate to address predefined research objectives. In addition, this method optimizes the particle release locations to minimize the requisite number of particles. Even after applying this method, the computational expense of IBM studies is still large. The model in chapter 3 seeks to increase the affordability of IBM studies by transferring some of the calculations to graphics processing units. Chapter 4 describes three algorithms that assist in interpreting IBM output by identifying coherent geographic clusters from population connectivity data. The first two algorithms have existed for nearly a decade and recently been applied separately to marine ecology, and we provide a direct comparison of the results from each. Additionally, we develop and present a new algorithm that simultaneously considers multiple species. Finally, in chapter 5, we apply these tools and a trait-based modeling framework to assess which species traits are most likely to impact dispersal success and patterns in the Gulf of Maine. We conclude that the traits influencing spawning distributions and habitat requirements for settlement are most likely to influence dispersal.

Thesis Supervisor: Rubao Ji
Title: Associate Scientist with Tenure

Acknowledgments

As with most endeavours, this dissertation would not have been possible without the support of many individuals. Although this is by no means a comprehensible list of those who have helped, a few are listed here.

From a scientific standpoint, each of the members of my Thesis Committee has always been willing to discuss the work and provide invaluable guidance. In particular, my Thesis Advisor, Rubao Ji, has been incredibly supportive throughout the entire graduate school experience. Numerous other Joint Program students and individauls at WHOI, particularly those who have passed through Ji's lab, have provided ideas for interesting avenues of research.

I have also been fortunate to have a local and supportive family and am grateful for that. The numerous meals from my parents, dog-sitting from Meghan, housing from JJ, and general support of Katie and Sandy has been invaluable in getting me through the process.

Financial support was provided by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program, Woods Hole Oceanographic Institution (WHOI) via the Ocean Ventures Fund (OVF), and the National Science Foundation through grant numbers OCE-1459133, 0928442, and 1031256. Computer time for the GPU modeling section was provided both by Massachusetts Institute of Technology and Chris Hill via access to the Massachusetts Green High Performance Computing Center and WHOI through the OVF grant. We thank Changsheng Chen for providing the FVCOM output.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Effectively managing marine resources in the face of climate change, resource exploitation, and other anthropogenic disturbances requires a thorough understanding and accurate description of the marine environment (Fogarty and Botsford, 2007). For marine fisheries, this description must include the geographic distribution of each species (Fogarty and Botsford, 2007). These distributions may vary widely in time due to fluctuations in the biotic and abiotic environment and between species that have different life history strategies (Cowen and Sponaugle, 2009). Although marine species exhibit a wide variety of life history strategies, many species share a common trait that broad scale geographic dispersal primarily occurs during a pelagic larval phase (Cowen and Sponaugle, 2009). This dissertation examines how the diversity of spawning, larval, and settlement traits influences larval dispersal in the Gulf of Maine. In doing so, it describes a suite of new methods for simulating larval dispersal, assessing the robustness of the results, and synthesizing the simulated patterns into a coherent message.

Population connectivity is defined as the intergenerational movement of individuals among geographically separated subpopulations and is a fundamental process in marine population dynamics (Cowen and Sponaugle, 2009). The scale and patterns of population connectivity help to determine the appropriate spatial scales for management, where management boundaries should be placed, and how robust the species will be to local and regional disturbances (Fogarty and Botsford, 2007; Cowen

and Sponaugle, 2009). For many marine species, population connectivity patterns are largely determined by the dispersal of planktonic larvae (Cowen and Sponaugle, 2009). Although dispersal patterns are predominately driven by ocean circulation patterns, species traits may exert considerable influence over them. Circulation patterns often vary in time, and so species that spawn continuously over the year may have substantially different connectivity patterns than species that only at a specific time or in response to a specific event (Cowen et al., 2007). Once in the water column, larvae may swim vertically or horizontally to influence their exposure to predators and prey or to influence their destination (Pineda et al., 2007). Later in life, larvae must find suitable habitat for settlement and survive recruitment into the juvenile or adult population (Pineda et al., 2007). Connectivity patterns emerge from the complex interactions among these processes, the physical environment, and survival to reproduction (Cowen et al., 2007; Pineda et al., 2007; Cowen and Sponaugle, 2009).

Field methods provide insight into larval dispersal patterns, but are generally limited in spatial resolution, temporal resolution, or level of detail. Direct methods of observation provide unambiguous evidence of dispersal trajectories, but are highly labor intensive and thus limited in the level of detail that is financially attainable. These methods include in situ observation of individual larvae in the field (Cowen and Guigand, 2008), using stable isotopes to mark the otolith of larval fish prior to dispersal (Thorrold et al., 2007), or identifying parent-juvenile offspring from genetic similarity (Planes et al., 2009). Indirect methods of observation instead infer population connectivity patterns from other variables and provide information about connectivity patterns over broad spatial scales, but with low spatial and temporal resolution. These methods may include inferring connections strengths from the genetic similarity among subpopulations (Lowe and Allendorf, 2010) or identifying the presence of natural geochemical markers in the otoliths of fish and calcified structures of invertebrates (Thorrold et al., 2007). Although observational methods of assessing larval dispersal patterns provide valuable insight, it is difficult to comprehensively summarize population connectivity patterns using them alone.

Biophysical models that couple Eulerian circulation models to Lagrangian particle-

tracking models provide a cost-effective method to estimate connectivity patterns with high spatial and temporal resolution and complement observational methods. Computing power, observation data, and our understanding of ocean dynamics have vastly increased over the past few decades, and ocean circulation models can now produce accurate, high resolution predictions of circulation patterns (Lynch et al., 2015). The output from these circulation models may then be used to force Lagrangian particle-tracking models, which numerically integrate the trajectories of many small particles as they move through the circulation fields. Individual-based models (IBMs) are an extension of Lagrangian particle-tracking models that allow individual traits to be prescribed to each particle. IBMs have been developed and are widely used to predict the dispersion of pollutants (e.g. Le Henaff et al., 2012), development of marine holoplankton (e.g. Ji et al., 2012), and dispersal of marine larvae (e.g. Mitarai et al., 2009; Paris et al., 2007; Petrik et al., 2014). Although Eulerian advection-diffusion models can and have been used as well for these same problems (e.g Cowen et al., 2000; Treml et al., 2008, 2012), there are a number of advantages to IBMs. Most prominently, IBM structure is easily and clearly relatable to the transport processes being modeled. Each individual in the IBM reacts to and changes its state in response to the local environment, and these changes remain with the individual for the duration of the model run. Some examples of processes that are often included in IBMs within marine ecology are growth and mortality in relation to predator and prey fields (e.g. Petrik et al., 2014), homing behavior of individual larva to appropriate settlement regions (e.g. Staaterman et al., 2012), or diel vertical migration (e.g. Churchill et al., 2011). Although Eulerian models may also include these same processes, there is necessarily an averaging that occurs every timestep within each grid cell. As a result, IBMs can better retain the stochasticity of small scale processes and complex interactions that occur throughout an individual's life than their Eulerian counterparts.

Although IBMs provide a vast amount of information about larval dispersal patterns at reasonable expense, their application to research questions does not come without difficulties. For the results of IBMs to be useful for marine management,

managers must be confident that the results from the IBM are robust and accurate. Accuracy must be assessed through comparison against observations, but robustness can be assessed through examination of the IBM results alone and is often achieved by simulating a large number of larvae. Chapter 2 presents a novel statistical method that simultaneously reduces the number of larvae required and assesses the robustness of the results. Even using this method, tens of millions of larvae or more may be required, and the simulation of these larvae may require thousands of computer hours. Chapter 3 presents an attempt to reduce this computational expense by simulating larvae on an alternative computing architecture. The high resolution trajectories that are produced by IBMs include a vast amount of information regarding Lagrangian transport, but they provide little understanding until distilled into more meaningful forms. Chapter 4 presents a new method to summarize IBM results in a meaningful way. Using the trajectories from an IBM to estimate the probabilities of transport among regions in the ocean, this method creates maps that depict coherent geographic clusters. In contrast to existing methods, it simultaneously considers multiple datasets that may represent different times or species and generates estimates of spatial structure, temporal dynamics, and species similarity. Finally, chapter 5 applies the methods from the prior chapters using a trait-based modeling framework to provide insight into the role of individual species traits in structuring dispersal patterns in the Gulf of Maine.

Chapter 2

Resource allocation for Lagrangian tracking

This chapter was previously published as Jones et al. (2016) and is governed by the below copyright policy.

Abstract

Accurate estimation of the transport probabilities among regions in the ocean provides

valuable information for understanding plankton transport, spread of pollutants, and movement of water masses. Individual based particle-tracking models simulate a large ensemble of Lagrangian particles and are a common method to estimate these transport probabilities. Simulating these particles is computationally expensive, and appropriately allocating resources can reduce the cost of this method. Two universal questions in the design of studies that use Lagrangian particle-tracking are how many particles to release and how to distribute particle releases. We present a method for tailoring the number and release location of particles to most effectively achieve the objectives of a study. Our method is a sequential analysis procedure that seeks to minimize the number of particles that are required to satisfy a predefined metric of result quality. We assess result quality as the precision of our estimates for the elements of a transport matrix, and also describe how our method may be extended for use with other metrics. Applying our methodology to both a theoretical system and a particle-transport model of the Gulf of Maine results in more precise estimates of the transport probabilities with fewer particles than from uniformly or randomly distributing particle releases. The application of our method can help reduce the cost of and increase the robustness of results from studies that use Lagrangian particles.

2.1 Introduction

Particle transport has implications throughout oceanography. Phytoplankton and zooplankton that form the base of the marine food web cannot overcome ocean currents and are transported as small particles (Miller and Wheeler, 2012). Higher trophic levels, including many invertebrates and fish, are transported as planktonic larvae (Pineda et al., 2007). Oil and other chemical pollutants often assemble into droplets that are transported as small particles (Lynch et al., 2015). Understanding the movement of these particles is critical to understanding marine ecosystems.

Our knowledge of particle transport may be represented as a connectivity matrix whose elements give the probability of transport among discrete geographic regions (Cowen and Sponaugle, 2009). One commonly used method to estimate connectivity matrices is to simulate many Lagrangian particles with an individual-based model (IBM) and compute the ensemble average of the particle trajectories. IBMs simulate particle transport through Eulerian velocity fields that are produced by ocean circulation models. Because some computational overhead is required to produce the Eulerian velocity fields, IBMs operate most efficiently when simulating large batches of particles. Each particle responds to its local environment based on the attributes that have been prescribed to it, which may include buoyancy, swimming behavior, growth, or other relevant processes (Irisson et al., 2009). This feature allows IBMs to be configured for a variety of particle types and has resulted in their use across multiple disciplines of marine science (Lynch et al., 2015).

Accurate predictions with IBMs are dependent on correct specification of the input parameters. In addition to individual particle attributes that may be estimated from field and laboratory data, IBM studies universally require that the researcher choose how many particles to release and how to distribute particles among multiple

origin sites. The number and distribution of particle releases regulates the tradeoff between computational time and result accuracy. Brickman and Smith (2002) present a discussion of the errors that may arise from releasing too few particles. The first type of error, which Brickman and Smith (2002) term U-I error, is that the number of particles is insufficient to capture the underlying statistics of the Eulerian velocity field. In the event of U-I error, an identically configured replicate trial will likely give different results. The second type of error, U-II error, is that the particle release distribution does not adequately sample a subarea of particular importance. When U-II errors occur, replicate trials with the same release locations will provide similar results, but the results do not accurately describe the properties of the region as a whole. Both Brickman and Smith (2002) and Simons et al. (2013) present methods to avoid these and similar errors. However, as we explain further in Section 2.5, the methods presented by Brickman and Smith (2002) and Simons et al. (2013) require that the researcher first simulate extra particles, then retrospectively identify how many particles would have been required. IBM studies may simulate tens of millions of particles and consume vast computational resources (e.g. Watson et al., 2012; Jones et al., 2015), and so we seek an alternative method that reduces the required number of particles.

The second design issue, how to distribute particles across origin sites, is more difficult and has been less thoroughly explored in existing literature. One option is to uniformly distribute releases across origin sites (e.g. Watson et al., 2012; Jones et al., 2015). In the case of ecological studies, an alternative is to distribute particle releases based on known spawning distributions (Gallego and North, 2009). However, knowledge of spawning distributions is often poor (Gallego and North, 2009). As we will show, the choice of release distribution may have substantial implications for the number of particles that are required for statistical confidence, and the issue of optimizing this release distribution is not addressed by previously published methods. We propose a sequential method to optimize the particle release distribution across the origin sites.

We demonstrate our innovative method by estimating the elements of a transport matrix. Our method addresses the following questions. First, how many particles must be simulated to robustly estimate the transport probabilities? Second, to minimize the number of particles required, how should particles be distributed across origin sites? Although our presentation is in the context of estimating the connection probabilities, the method may be applied to other objectives, such as parameterizing models of population dynamics or assessing the contamination risk from pollutant spills. In addition to the description of our method here, we are also releasing multiple software packages that implement it.

2.2 A sequential Bayesian procedure

Consider a study system with n_o origins and n_d destinations. Let p_{ij} be the unknown probability that a particle released from origin i is at destination j at a specified time and let $P = [P_{ij}]$ be the n_oxn_d matrix of these probabilities (Table 2.1). Our

goal is to estimate P to a specified precision using a minimal number of particles. Under the sequential Bayesian approach proposed here, the matrix P is treated as a random variable. Throughout our description of this procedure, we follow the common statistics convention that random variables are indicated by capital letters (e.g. P_{ij}), and realizations of these variables by lowercase letters (e.g. p_{ij}). As described in more detail below, at each step of the sequential procedure, the current value of an objective function measuring estimation precision is compared to a stopping criterion (Figure 2-1). If the criterion is met, the procedure terminates and each element of P is estimated by its current expected value. If the criterion is not met, the current distribution of P is used to allocate a new batch of particles to the origins, these particles are released, the current distribution of P is updated based on their observed destinations, and the procedure is repeated. In this section, we describe the basic statistical model, the stopping criterion, and the allocation rule.

Statistical model

Let $m_i^{(k)}$ be the number of particles through step k of the sequential procedure that have been released from origin i and let the random variable $X_{ij}^{(k)}$ be the number of these with destination j. Under the assumption that the destinations of different particles are independent and conditional on $p_i = (p_{i1},\ p_{i2},\ \ldots,\ p_{i,n_d})$, the vector $X_i^{(k)} = (X_{i1}^{(k)},\ \ X_{i2}^{(k)},\ \ldots,\ X_{i,n_d}^{(k)})$ has a multinomial distribution with $m_i^{(k)}$ trials and probability vector p_i with probability mass function given by Equation 1. The probability mass function below describes the likelihood of observing any realization, $x_i^{(k)}$, of the random variable $X_i^{(k)}$.

$$pr(x_i^{(k)}|p_i) \propto \prod_{j=1}^{n_d} p_{ij}^{x_{ij}^{(k)}} \tag{2.1}$$

To implement the Bayesian approach, it is necessary to specify a prior distribution for the probability vector P_i. A natural choice is the Dirichlet distribution with probability density function:

$$pr\,(p_i) \propto \prod_{j=1}^{n_d} p_{ij}^{\alpha_{ij}-1} \tag{2.2}$$

with parameters $\alpha_{i1},\ \alpha_{i2},\ \ldots,\ \alpha_{i,n_d}$. In the absence of prior information, it is again natural to take $\alpha_{ij} = 1$ for all i and j so that all possible values of P_i are equally likely. It follows that the distribution of P_i after step k is itself Dirichlet with updated parameters $\alpha_{ij}^{(k)} = 1 + x_{ij}^{(k)}$. This reflects the fact that the Dirichlet distribution is the conjugate prior distribution for multinomial data.

Stopping criterion

At step k, for each origin i, the current distribution of P_i is Dirichlet with parameters $\alpha_{\mathrm{ij}}^{(k)} = 1 + x_{\mathrm{ij}}^{(k)}$, $j = 1,\ 2,\ \ldots,\ n_d$. The decision whether to terminate the procedure

Symbol	Description
n_o	The total number of origin sites where particles are released.
n_d	The total number of destination sites where particles may arrive.
$P = [P_{ij}]$	The connectivity matrix. p_{ij} is the unknown probability that a particle released from origin i will arrive at destination j. P is the matrix of these probabilities, and the random variable P_i is the i^{th} row of P.
p_i	A single realization of the random variable P_i.
$m_i^{(k)}$	The number of particles that have been released from origin i up to and including step k.
$X_{ij}^{(k)}$	A multinomially distributed random variable representing the number of particles released from origin i that arrive at destination j up to and including step k of the procedure. The vector $X_i^{(k)} = (X_{i1}^{(k)}, X_{i2}^{(k)}, \ldots, X_{i,n_d}^{(k)})$.
$x_{ij}^{(k)}$	A single realization of the random variable $X_{ij}^{(k)}$ that gives the observed number of particles released from origin i and arriving at destination j up to and including step k of the procedure.
$\alpha_i^{(k)}$	The vector of parameters for the Dirichlet distribution for P_i at the end of step k. $\alpha_i^{(k)}$ is composed of $\alpha_{i1}^{(k)}, \alpha_{i2}^{(k)}, \ldots, \alpha_{i,n_d}^{(k)}$.
$\mu_{ij}^{(k)}$	The mean of the Dirichlet distribution for P_i after step k.
$\sigma_{ij}^{(k)}$	The standard deviation of the Dirichlet distribution for P_i after step k.
$CV_{ij}^{(k)}$	The coefficient of variation of the Dirichlet distribution for P_i after step k.
$H^{(k)}$	The objective function used to determine when to terminate sampling and how to allocate particle releases.
δ	A threshold that determines when p_{ij} are too small to be relevant to the study goals.
π	A probability threshold that determines when p_{ij} are too small to be relevant to the study goals.
ϵ	The threshold value for $H^{(k)}$ that determines when sampling terminates.
b	The number of particles simulated in each batch.

Table 2.1: The parameters for our sequential analysis routine are collected and defined here. Following common statistics convention, random variables are indicated with capital letters and realizations of these variables are indicated with lower case letters.

and estimate p_{ij} by its current mean:

$$\mu_{ij}^{(k)} = \frac{\alpha_{ij}^{(k)}}{\sum_{j=1}^{n_d} \alpha_{ij}^{(k)}} \tag{2.3}$$

or to release additional particles must be made on the basis of this distribution. One measure of the current uncertainty in P_{ij} is its coefficient of variation:

$$\mathrm{CV}_{ij}^{(k)} = \frac{\sigma_{ij}^{(k)}}{\mu_{ij}^{(k)}} \tag{2.4}$$

where:

$$\sigma_{ij}^{(k)} = \sqrt{\frac{\alpha_{ij}^{(k)} \left(\sum_{j=1}^{n_d} \alpha_{ij}^{(k)} - \alpha_{ij}^{(k)} \right)}{\left(\sum_{j=1}^{n_d} \alpha_{ij}^{(k)} \right)^2 \left(\sum_{j=1}^{n_d} \alpha_{ij}^{(k)} + 1 \right)}} \tag{2.5}$$

is the current standard deviation of P_{ij}. We take as a measure of overall precision the objective function:

$$H^{(k)} = \max_{i,j} \left(CV_{ij}^{(k)} : pr^{(k)} \left(P_{ij} > \delta \right) > \pi \right) \tag{2.6}$$

where $pr^{(k)}(P_{ij} > \delta)$ is the current probability that P_{ij} exceeds δ. δ and π are small user-specified probabilities. The side condition is required because $CV_{ij}^{(k)}$ becomes excessively large if the current distribution of P_{ij} is concentrated near 0. The procedure terminates when $H^{(k)}$ first falls below a specified value ϵ. The choice of the constants δ and π is discussed below in Section 3.6.

Allocation rule

If the stopping criterion is not satisfied in step k, step $k+1$ begins by sequentially allocating each of a batch of b particles to an origin site. Consider allocating the first such particle under the assumption that, for each origin, the destination of this particle is known. For each origin, we would update the current distribution of P to include this particle via Bayes' Theorem, compute the value of the objective function $H^{(k)}$, and allocate the particle to the origin for which the value of $H^{(k)}$ is smallest. In practice, the destination of the particle released at a particular origin is unknown until the entire batch has been allocated and the IBM has been run. For this reason, the particle is allocated to the origin with the smallest expected value of the stopping criterion, where this expected value is computed by integrating over the entire predictive distribution for the destination. For a single particle released from origin i, this predictive distribution is Dirichlet-multinomial with 1 trial and parameters $\alpha_{ij}^{(k)}$, $j = 1,\ 2,\ \ldots,\ n_d$.

A simulation approach to approximating the expected value of the stopping criterion for a single particle released from origin i proceeds as follows. Simulate a realization p_i^* of P_i from the Dirichlet distribution with parameters $\alpha_{ij}^{(k)}$, $j = 1,\ 2,\ \ldots,\ n_d$. Simulate a destination from the multinomial distribution with 1 trial and probability vector p_i^*. Update the current distribution of P_i based on this simulated destination and compute the new value of the stopping criterion. Repeat the process many times and approximate the expected value of the stopping criterion by the average of its new values generated from these simulated destinations.

The same general approach is used to allocate the second particle except that destinations are simulated for both the first and second particles. However, in allocating the second particle, the origin of the first particle remains fixed at the origin selected as described above. The process is repeated for each particle in the batch. Because the origins of previously allocated particles are not reconsidered when allocating later particles, this procedure is not guaranteed to identify the optimal allocation of the batch of particles. Pseudocode to implement this allocation rule is provided in Appendix A.

2.3 Validation using artificial data

We validated our procedure using artificial data based on ecological networks (e.g. Kininmonth et al., 2010; Watson et al., 2011; Jones et al., 2015). For each replicate, we constructed a connectivity matrix, then drew multinomial samples from it that represent Lagrangian particles. Because we know the underlying connectivity matrix, this test ensures convergence to the correct solution.

Our objective function measures the precision of each p_{ij}, which may also be measured by the percent error in the estimated connectivity matrix when the true connectivity matrix is known. Because the connectivity matrix that was used to generate the artificial data is known, the artificial data may be used to assess the relationship between the objective function $H^{(k)}$ and the percent error. We randomly generated 25 connectivity matrices with each of $n_o = (4, 9, 16,$ and $25)$ origins and $n_o + 1$ destinations. The first n_o destinations were the same as the origins, and between 0% and 10% of the particles returned to these origins. Destination n_d represented everywhere else. Each row of these connectivity matrices gives the probability vector for a multinomial distribution from which we took samples that represent Lagrangian particles. We treated these samples as a single run of a Lagrangian particle tracking model and estimated the connectivity matrix, then computed $H^{(k)}$ from this estimate. $H^{(k)}$ provides an upper bound on the percent error (Figure 2-2), indicating that it is a valuable error metric. The value of $H^{(k)}$ is inversely related to the number of particles that followed each possible pathway. When few particles have been simulated relative to the number of destinations, $H^{(k)}$ is large, indicating that these few particles may not provide a good estimate for the connectivity matrix. However, as the number of particles increases, both $H^{(k)}$ and the percent error decrease, and so small $H^{(k)}$ correctly indicates that the percent error is small. Fewer particles are required for connectivity matrices with fewer destinations because having fewer destinations results in larger

transport probabilities under our connectivity matrix generating scheme. Although the expected value of the posterior percent error could have been used instead of the CV based objective function, the CV has the practical benefit of an analytic solution and accurately indicates when the percent error is small.

We also tested that the allocation rule results in faster convergence of $H^{(k)}$ than either uniformly or randomly distributing particle releases. The uniform distribution represents the null case where particles are released throughout the domain, and the random distribution mimics particle releases based upon criteria that do not correlate well with the flow patterns (e.g. species distributions). Our method consistently outperformed both alternatives in 10 simulations, and the simulations revealed interesting aspects of $H^{(k)}$ (Figure 2-3). The objective function initially reacts only to the missing connections which have the largest CV, and $H^{(k)}$ reduces to $\sqrt{m_i^{(k)}(m_i^{(k)}+2)^{-1}}$ for these connections. Therefore, the objective function initially increases asymptotically towards 1 until these missing connections are identified, then subsequently decreases. Because our allocation rule assumes that the objective function monotonically decreases as more particles are simulated, this property of the objective function is problematic. The threshold number of particles required to satisfy $P(p_{ij} \geq \delta) < \pi$ may be computed by solving the relation $\pi = F(\delta, 1, n_i)$, where $F(\delta, 1, m_i^{(k)})$ is the cumulative distribution function of the beta distribution with shape parameters 1 and $m_i^{(k)}$ evaluated at δ, and we recommend that users release this number of particles from each origin in the first batch. Once the missing connections are identified, the allocation rule outperforms the alternatives, and $H^{(k)}$ decreases in proportion to the square root of the number of particles. $H^{(k)}$ may also increase when p_{ij} are approximately equal to δ. In this scenario, connections alternate between satisfying and not satisfying $P(p_{ij} \geq \delta) \geq \pi$, and rapid changes in the value of $H^{(k)}$ occur as shown by the uniform allocation scheme in Figure 2-3. However, these changes are transient features, and so the allocation rule performs well in spite of them. In all trials, the random distribution resulted in poor convergence of the objective function, suggesting that allocation schemes based on spawning distributions should be avoided when the objective is to precisely estimate the connectivity matrix.

Overall, our method performs well on artificial networks that represent ecological networks. It converges to the correct solution, and converges more quickly than either null distribution of particle releases.

2.4 Validation using a realistic tracking simulation

We further validated our method using a simulation of the Gulf of Maine as a representative IBM study. Our simulation is based upon that of Huret et al. (2007). For brevity, we describe only where our study differs from the original. We used a particle tracking model to simulate cod larval dispersal during January 1995. We forced the particle-tracking model with hourly output from the Finite-Volume Coastal Ocean Model (FVCOM, Chen et al., 2003). FVCOM was configured using the 3^{rd} generation of the Gulf of Maine mesh, which contains 48451 nodes and 90415 elements that

smoothly transition from 200 m resolution at the coastline to 15 km resolution in the central Gulf of Maine and extends from Maryland to Nova Scotia (Figure 2-4).

Particles that represent cod larvae were released from three spawning sites along the coast of New England (Figure 2-4) throughout January 1995. The spawning grounds were taken from the map published in Huret et al. (2007). Particle release locations within each spawning region were randomly selected in time and space from a uniform distribution. Particle destinations were computed from the position of the particle at 60 days age.

Our first test validated the use of a multinomial distribution. The multinomial distribution assumes independence between particles, which may not be appropriate if particles are released closely in space and time. We released 1000 particles from each spawning ground and estimated the connectivity matrix. We repeated this process 100 times with different release locations and timing and obtained 100 estimates for each element of the connectivity matrix. We assumed that the mean of these 100 trials represents the expected outcome, and tested this assumption using the variance test from Brickman and Smith (2002). The variance of the mean leveled off after 40-60 trials, indicating that our use of 100 trials is sufficient (Figure 2-5). We computed the χ^2-statistic for each element, $\sum_{k=1}^{100} (p_{ij} - \hat{p}_{ij}^{(k)})^2 p_{ij}^{-1}$, where $\hat{p}_{ij}^{(k)}$ is the estimate of p_{ij} from the k^{th} trial and p_{ij} is the mean of these estimates across all 100 trials. The observed distributions of the χ^2-statistics did not differ from those that would result from multinomial sampling (Figure 2-6).

Our second test evaluated the allocation rule. We sequentially released batches of 500 particles whose distribution was determined by our allocation rule, by a uniform distribution, or by a randomly chosen distribution until our computational budget of 50,000 particles was exhausted. During these tests, we set $\epsilon = 0.1$, $\delta = 0.005$, and $\pi = 0.05$. In 3 repetitions, our methodology consistently increased the convergence rate of $H^{(k)}$ (Figure 2-7). Only the optimized distribution scheme satisfied the stopping criterion within the budget by reaching the threshold value of 0.1. Upon exhausting the budget, $H^{(k)} = 0.11 \pm 0.0039$ (mean $\pm$ std. dev.) for uniformly distributed particles and $H^{(k)} = 0.28 \pm 0.030$ for the random distribution. The optimized distribution satisfied the stopping criterion after simulating 26,666$\pm$3,253 particles. At the point where the optimized distribution satisfied the stopping criterion, $H^{(k)} = 0.14 \pm 0.008$ for the uniform distribution and $H^{(k)} = 0.40 \pm 0.017$ for the random distribution. A detailed presentation of this evaluation, including source code, is presented in Appendix B.

2.5 Alternative methods

Choosing the number of Lagrangian particles is a fundamental component of IBM studies, and previous publications have described alternative methods to address this issue. Brickman and Smith (2002) proposed the variance test as a method to identify the presence of both U-I and U-II errors. To apply the variance test, researchers first generate a set of release locations that evenly distributes b particles throughout a sin-

gle origin site. The researchers then perform t replicate simulations using this release distribution. Variability among the trials emerges due to a stochastic component in the particle velocities, and this variability is quantified with the test statistic, $V^{(k)}$. To compute $V^{(k)}$, the researchers draw a random sample of k trials from the t trials available. $V^{(k)}$ is the mean variance in a sample of size k divided by k. $V^{(k)}$ decays with increasing k, and the researchers may be confident that their results are not subject to U-I error when the $V^{(k)}$ vs. k curve levels off. To protect against U-II error, they suggest modifying the variance test to use increasing b instead of increasing k.

Simons et al. (2013) propose an alternative method to test the related question: how many particles are required to ensure that a simulation closely approximates a reference solution? The first step in their method is to compute a single large trial with b particles and compute a reference solution. Because this solution is computed from the largest number of particles available, they assume that it provides the best representation of the underlying flow and seek to replicate it with a reduced number of particles. They begin by drawing a random subset of s particles from the pool or b particles, and compute a sample solution from this subset. They then compare the sample solution to the reference solution by computing the fraction of unexplained variance (FUV) between the solutions as $\text{FUV}^{(s)} = 1 - r^2$, where r is the linear correlation coefficient between the solutions. Repeating this process many times for multiple values of s, they obtain a curve that plots $\text{FUV}^{(s)}$ against s. Finally, they threshold this curve when $\text{FUV}^{(s)}$ is sufficiently small to identify an appropriate value of s.

Although our procedure, the variance test, and the FUV method all address similar questions, our method is structured differently from the others in order to reduce the required number of particles. Both the variance test and FUV method begin by simulating a large pool of trials or particles, then subsample from this pool to estimate the variability in the results. For the variance test, t must be greater than k to subsample and compute $V^{(k)}$. For the FUV method, b must be greater than s to estimate $\text{FUV}^{(s)}$. In contrast, our method alternates between simulating particles and assessing convergence, then terminates as soon as convergence is achieved. However, this design choice prohibits subsampling from a larger pool to estimate the variability in the results, and instead we estimate the variability from the properties of the posterior distribution for each p_{ij}. Each of the three proposed methods has merits in addressing issues related to the number of required particles for IBM studies, but differs slightly in how each does so.

2.6 Discussion

We provide a flexible and reliable method to match particle release counts and distributions to the specific objectives of a particular study. The method avoids both U-I and U-II errors discussed in Brickman and Smith (2002). U-I errors occur when replicate simulations would result in substantially different results. Our method avoids this error by evaluating a stopping criterion and continuing the simulation until variability in the results is sufficiently small. U-II errors occur when the release distribution

skips over subregions of particular importance. Whereas Brickman and Smith (2002) evenly distribute particle releases throughout each origin and reuse the same release locations for each trial, we draw a new set of release locations from a uniform distribution for each step. This procedure avoid U-II errors altogether, because a large number of randomly drawn points will represent the underlying structure of each origin. Although we draw the release locations within each origin from a uniform distribution in our examples, egg production models or fine-scale field data may be used to generate these distributions when such information is available (Gallego and North, 2009). Our method also addresses how to distribute releases among multiple origins in order to minimize the number of particles required to achieve statistical confidence, which has not been done by prior studies.

Although our method assumes that b particles are simulated in each batch, choosing b is dependent on the specific IBM being used. IBMs may be operated in online mode and load the Eulerian velocity fields directly from a hydrodynamic model, or in offline mode and read the velocity fields from archived output of a hydrodynamic model. In either case, there is a computational cost to operating the hydrodynamic model or reading the circulation fields. This cost is incurred every time a batch of particles is simulated, but is largely independent of the number of particles being simulated in each batch. A tradeoff emerges where small b allows our method to most effectively allocate particles among origins and terminate most quickly, but large b increases the efficiency of the IBM and reduces the cost per particle. Choosing an optimal value of b may reduce the computational cost required to achieve convergence, but the choice of b does not influence when our method deems that convergence has been achieved. The computational overhead of loading the velocity fields is specific to each IBM and hydrodynamic model configuration, and so we recommend that researchers choose b such that their IBM operates with a reasonable level of efficiency.

Our method also assumes that the multinomial distribution is an appropriate model for the particle destinations, which implies that the trajectories are independent. Multiple releases that are closely located in time and space may result in correlation among trajectories. However, randomly chosen release locations within an origin, releases separated by at least the velocity decorrelation scale, or tracking durations longer than the Lagrangian decorrelation time will likely avoid this concern. Each particle may only contribute to one destination, which excludes settlement criteria based on the proportion of time that a particle spends within a destination region (e.g. Huret et al., 2007). An alternative is to assign each particle a probability of settling during each timestep, then remove it from further consideration after settlement (e.g. Tian et al., 2009b).

Our examples focus on a single objective function and stopping rule that reflect our objectives from applying this procedure. Because the CV responds to the uncertainty in each p_{ij} relative to the value of that p_{ij}, it is appropriate for use when the estimates for p_{ij} are multiplied together and errors would be multiplicative (e.g. in a matrix projection population model). Likewise, ignoring very small p_{ij} was chosen to reflect that very low connectivity rates among subpopulations may not substantially impact population demographics (Hastings, 1993; Lowe and Allendorf, 2010). Choosing the parameters δ and π for this objective function is study specific, but here we present

some examples for consideration. In ecology, only a few migrants per generation are necessary to maintain genetic homogeneity, and many fish spawn millions of eggs each year (Slatkin, 1987). Therefore, studies examining genetic connectivity must quantify even rare connections, and $\delta = 10^{-6}$ may be appropriate. However, more frequent exchange of individuals is required for connectivity to influence population dynamics, and so studies examining population demographics may set $\delta = 10^{-2}$ (Hastings, 1993; Lowe and Allendorf, 2010). The second parameter, π, is analagous to the significance level in frequentist statistical tests, and so we suggest $\pi = 0.05$ as a default value. However, these are merely default suggestions, and researchers may alter them based upon the goals of individual studies.

More broadly, users may replace Equation 2.6 with an appropriate representation of what is important in their system. The objective function must take the parameter vectors α as an argument and return a single scalar value that quantifies the quality of α. For example, ecological studies that include population connectivity as one component of a population model may quantify the variability of the results differently. Realized population connectivity patterns include spawning distributions and postsettlment survival (Watson et al., 2010). Researchers seeking to estimate these patterns may develop a population model that includes these processes, evaluate the population model using many credible values for the connectivity matrix, P, and seek to minimize the variance in the evaluations. Either the output of the particle-tracking model or the objective function must include all processes relevant to the study, including, for example, survival and growth of larvae and loss of particles to the model boundaries. The allocation rule relies on two assumptions that any choice of objective function must satisfy. First, the objective function must decrease as the quality of the estimated connectivity matrix increases. Second, releasing more particles from an origin must reduce the contribution of that origin to the value of the objective function. We suggest that practioners test these assumptions when using a new objective function. The software package associated with this publication includes methods for performing this test. Our allocation rule is a greedy heuristic that provides an improved, but suboptimal, particle distribution. In the future, we hope to provide theory that bounds the difference between the output of our allocation rule and the optimal solution.

Particle counts in particle tracking studies vary widely from a few thousand (e.g. Huret et al., 2007; Tian et al., 2009a) to tens of millions (e.g. Watson et al., 2012; Jones et al., 2015). Field research that relies on parentage, tagging, or drifter data may be limited to only a few hundred sample points (Almany et al., 2007; Planes et al., 2009). The appropriate number of particles is dependent on the study goals, and readers and authors must take care to avoid drawing conclusions beyond those that can be justified by the number of particles. Our method provides a robust and quantitative way to determine the count and distribution of particle releases, which can help researchers to obtain more precise estimates of transport probabilities with reduced costs, draw appropriate conclusions from tracking experiments, and thus better understand marine ecosystems.

2.7 Code availability

An online interface to our method is available at `http://btjones.scripts.mit.edu/index.fcgi/research/sequential-analysis-method`. Source code and instructions for installing and accessing our method is available from `https://github.com/btjones16/sequential-analysis-software`. The source code repository includes R and C++ libraries, together with a SWIG interface file that allows access to the C++ library from Python, Octave, and other scripting languages (Beazley, 1996).

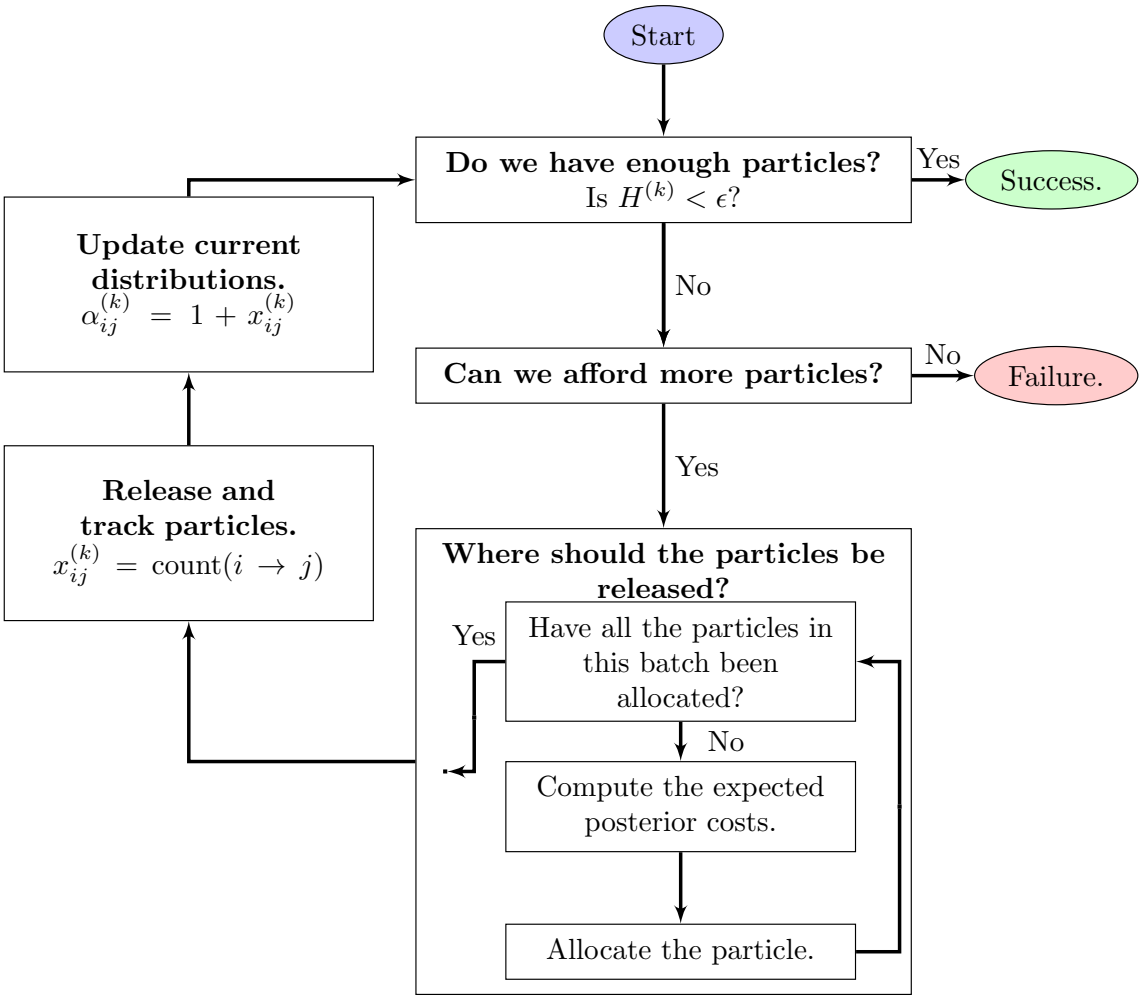

Figure 2-1: The sequential analysis procedure is an iterative process. Each iteration, it first assesses if enough particles have been simulated based on the stopping rule. If not and if additional particles are within the computational budget, then the particles are distributed according to the allocation rule. If at any time the stopping rule is satisfied or the budget is exhausted, the procedure is terminated with either a successful or failed result.

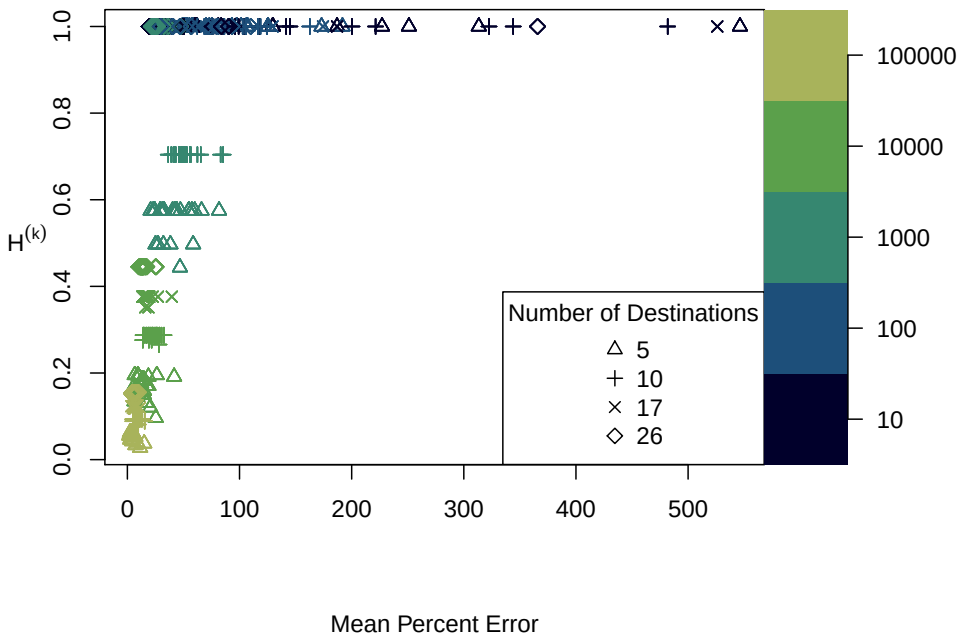

Figure 2-2: The objective function (vertical axis) is plotted against the mean percent error in the estimated connectivity matrix (horizontal axis). Each data point was computed by randomly generating a matrix $x^{(k)}$ from one of the artificially generated connectivity matrices. The color indicates the number of particles that were included in $x^{(k)}$, and the plotting symbol indicates the number of destinations in the connectivity matrix.

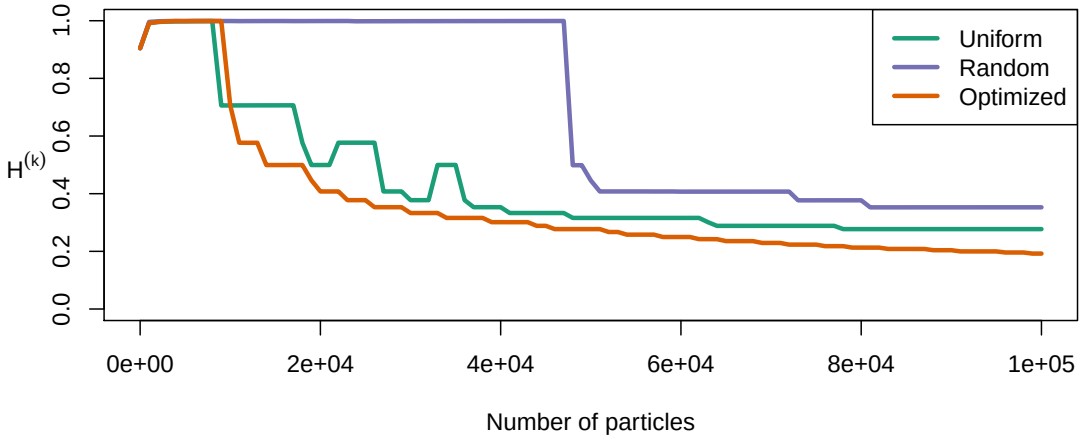

Figure 2-3: Ten sequential simulations were run using 9 node artificially generated connectivity matrices. The results of all ten were similar, and so only 1 of them is plotted here. The number of particles included for the estimate for $H^{(k)}$ is depicted on the horizontal axis, and the particle allocation scheme is given by the color of the line.

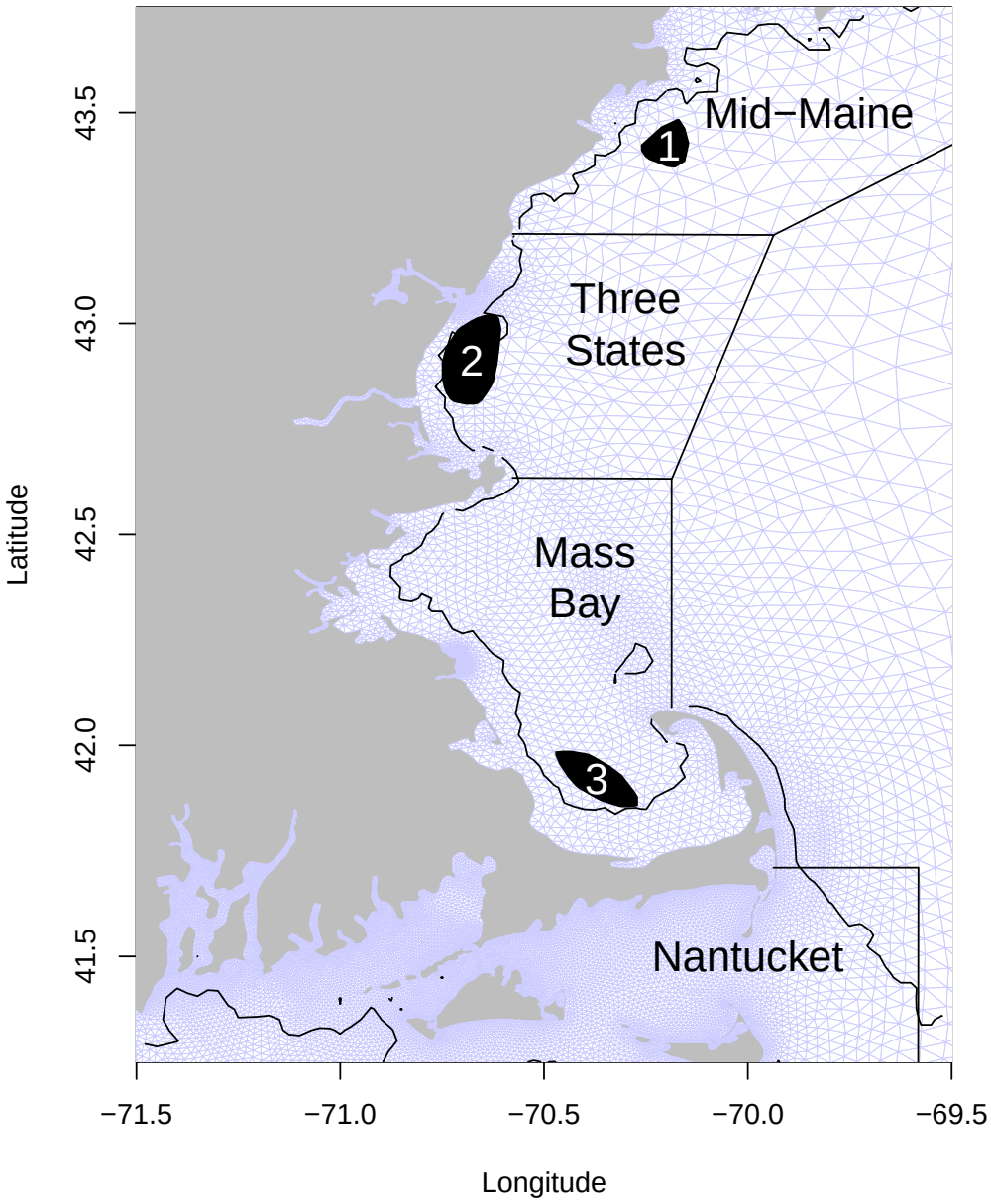

Figure 2-4: The study regions are depicted here. The numbered sites are the particle release locations. The straight boundary lines indicate the destination regions, and the black line nearshore indicates the 30m isobath that was used to determine suitable habitat. The blue background mesh is the FVCOM mesh.

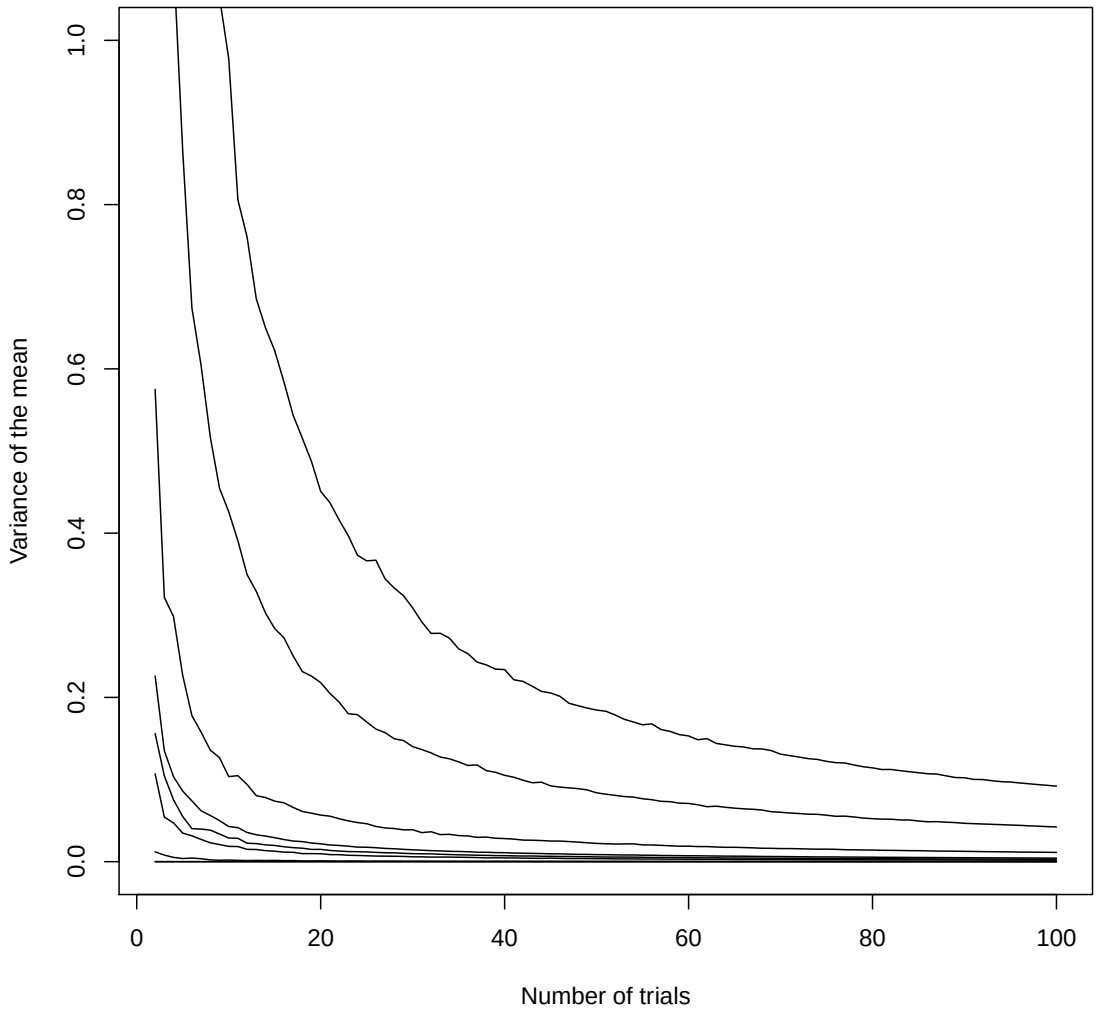

Figure 2-5: The variance of the mean estimate for each p_{ij} is plotted as a function of the number of trials included in the estimate. Each line represents one of the 12 p_{ij} in the connectivity matrix that we estimate in Section 4. This figure was constructed using the method described for the variance test in Brickman and Smith (2002) with 250 subsamples being drawn for each data point.

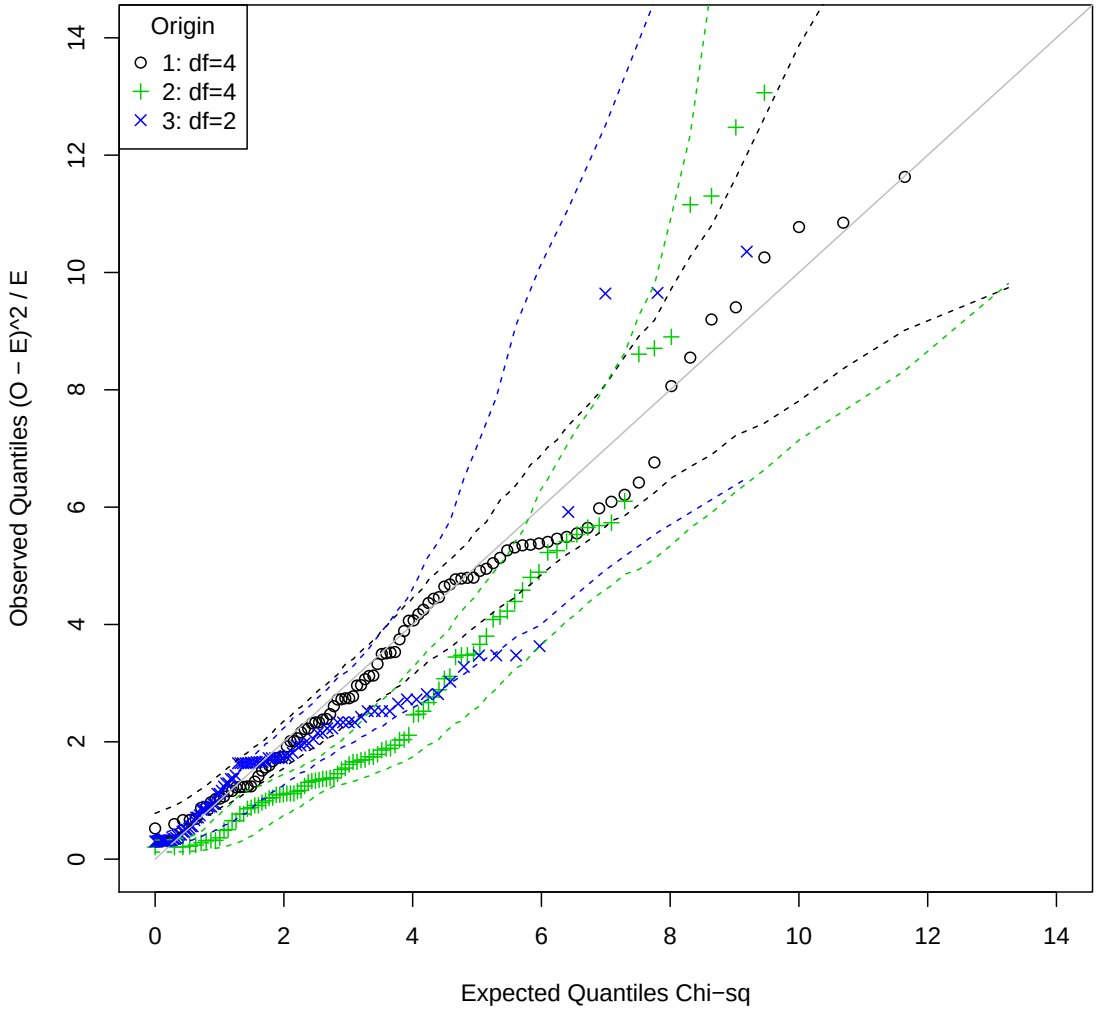

Figure 2-6: The expected quantiles from a Chi-squared distribution are plotted against the observed quantiles of the Chi-squared statistic from many particle-tracking simulations. The dashed lines indicate a 95% confidence interval, and the solid line indicates a one-to-one relationship. For origins 1 and 2, we observed 5 possible destinations, and so there are 4 degrees of freedom in the Chi-squared distribution. For origin 3, particles only went to three destinations due to strongly directional southern flow, and so there are only 2 degrees of freedom.

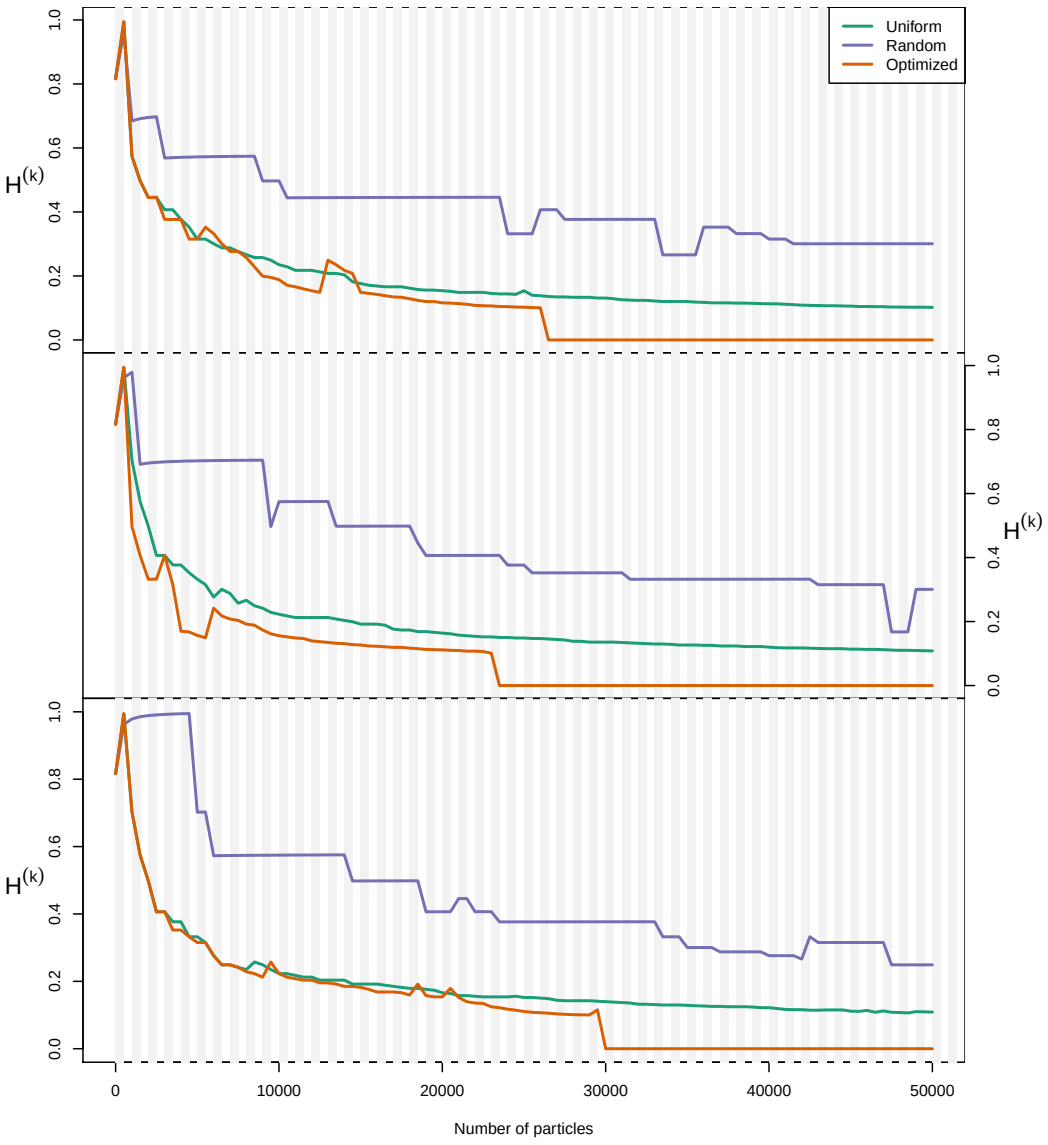

Figure 2-7: Particles were released particles uniformly, randomly, and using the allocation rule 3 times in a particle-tracking model for the Gulf of Maine. Particles were simulated in batches of 500, which are indicated by the shaded regions, and a total budget of 50,000 particles was permitted. The colored lines display the decrease in value for the objective function during each simulation and under each particle release scheme.

Chapter 3

A CPU and GPU capable Lagrangian particle-tracking model

Abstract

Describing the movement of Lagrangian particles in the ocean is a crucial component of describing ocean currents, developing response plans to pollutant spills, and understanding marine larval dispersal patterns. Individual-based models (IBMs) that track Lagrangian particles as they move through Eulerian circulation fields are a common method to simulate particle transport in the ocean. Because IBMs rely on the ensemble average of millions of particles, computational performance is a paramount concern in IBM design. Although other scientific applications have benefited by performing some of the calculations on graphics processing units (GPUs), most current IBMs execute solely on more traditional central processing units (CPUs). We present here a new IBM that has been designed and optimized for performance on both GPUs and CPUs. Averaged throughout the entire run for a representative configuration, we find that operating the model on the GPU is 2.15x faster than on the CPU alone. However, the performance benefits are unevenly distributed throughout the model, and certain procedures are hundreds of times faster on the GPU than the CPU. We discuss why this uneven distribution of performance benefits emerges, what implications it may have for IBM performance on GPUs and CPUs, and how the performance of our IBM could be further improved. We conclude with the recommendation that IBMs execute on both CPUs and GPUs for optimal performance.

3.1 Introduction

Accurately describing the movement of plankton, pollutants, and other materials in the ocean is an important component of effectively managing marine resources and responding to disasters (Lynch et al., 2015). However, in many cases, the nature of the material being transported or the desired management objective inhibits using field observations alone to describe transport patterns. For instance, the geographic spread of many marine species is regulated by a short, planktonic, larval phase, but

the broad geographic scale, small size of each individual, and high mortality rates render comprehensive field sampling prohibitively expensive (Cowen and Sponaugle, 2009). In the case of contaminants, response plans to contamination events must be developed in advance, so methods to estimate the likely dispersal trajectories prior to a contamination event are necessary. In each of these cases and others, numerical models provide a cost effective and feasible method to explore transport processes.

Methods to model and describe ocean circulation and transport patterns may be broadly divided based on their representation of the physical environment. Eulerian descriptions use a coordinate system that is fixed relative to the Earth, and the data often consist of repeated observations of the environmental state at predetermined geographic coordinates. In contrast, Lagrangian descriptions use a coordinate system that is fixed relative to the water, so the data are observations of the same water parcel as it moves through time and space. Eulerian descriptions of the ocean environment are analogous to a grid of moored buoys, and mesoscale and global scale hydrodynamic models generally use an Eulerian coordinate system due to computational considerations (e.g. Bleck et al., 2002; Shchepetkin and McWilliams, 2005; Chen et al., 2006). However, many materials that are transported through the ocean are impacted by their local environment, and Lagrangian descriptions of their movements may be better able to capture important local scale processes (Grimm and Railsback, 2005; Lynch et al., 2015). Particle-tracking models that track the position of Lagrangian particles as they move through an Eulerian circulation field are an effective way to explore transport processes in the ocean (Lynch et al., 2015). Individual-based models (IBMs) extend particle-tracking models to simulate not only the location of each individual in time and space, but also biological traits such as age and swimming behaviors (Grimm and Railsback, 2005).

IBM studies often rely on the ensemble average of millions of individuals to describe transport patterns, and simulating these individuals requires efficient utilization of computational resources (e.g. Watson et al., 2012; Jones et al., 2015; Treml et al., 2015). Although existing IBMs differ in the details of their implementation and the features available to users, the computational challenges facing IBM developers and the solutions to them are highly similar across oceanographic IBMs. Broadly, the computational challenges facing IBM developers are to quickly load snapshots of the circulation patterns, interpolate the circulation fields to the exact time and location of each individual, and advance the state of each individual. Oceanographic IBMs commonly address the first challenge by running in offline mode, where an Eulerian circulation model is first run to completion, and then the archived output is read back into the IBM (e.g. North et al., 2011; Ji et al., 2012; Paris et al., 2013). In contrast to online mode, where the circulation model and IBM are linked in real-time, offline mode permits multiple IBM scenarios to be run without the computationally intensive task of rerunning the circulation model. However, the high resolution circulation datasets may contain millions of velocity observations for each snapshot, and efficiently loading these snapshots demands high performance file formats and hardware. The second and third challenges are primarily related to the speed at which computations may be performed, and addressing them demands the use of high performance computing hardware together with efficient code. To meet these demands,

IBMs have traditionally been designed as Fortran programs that operate on Linux based clusters of central processing units (CPUs) and rely on binary NetCDF files for data storage (North et al., 2011; Paris et al., 2013). Although models following this design have been successfully applied to scientific problems for decades and continue to support cutting-edge oceanographic research, alternative designs may be better suited to individual based modeling on modern computing platforms. The software presented in this chapter is an alternative to the traditional IBM design, and the performance analyses support our hypothesis that this alternative design is well suited to conducting research in a modern computing environment.

Transitioning scientific models to graphics processing units (GPUs) has recently garnered attention as a method to improve computational performance (Lee et al., 2010; Herault et al., 2010; Couturier, 2014). Although the two principal processing units within a computer, CPUs and GPUs, are similar in many ways, they have been optimized for different tasks. CPUs have largely been optimized to reduce latency, which is the lag between when a operation is requested and when the CPU completes that operation (Owens et al., 2008; Couturier, 2014). Incredible feats have been achieved to make CPU cores smaller and faster over the past few decades, but the clock speed of individual CPU cores has largely stagnated in the past decade due to limitations on heat dissipation and transistor sizes (Moore, 1998; Sanders and Kandrot, 2010). Partially in response to these limitations, engineers have packed multiple cores onto a single chip to create parallel processors that can simultaneously complete multiple tasks. GPUs take this approach to an extreme and pack hundreds or thousands of relatively slow cores into each GPU to optimize bandwidth, or the volume of data that can be processed (Sanders and Kandrot, 2010; Couturier, 2014). Whereas CPUs are ideally suited for tasks that require quick responses to unknown commands (e.g. processing keystrokes from a user), GPUs are well suited to performing predefined tasks quickly on large amounts of data (e.g. rendering videos). Because IBMs perform the same relatively simple tasks on many pieces of data to advance the state of millions of particles, there is good reason to believe that they will run efficiently on GPUs. However, to the best of our knowledge, there is no published implementation of a GPU-based IBM for ocean modeling, nor is there a published description of the challenges and solutions to implementing one.

This chapter and the associated model seek to fill this void in three ways. First, we hope to present a summary of the challenges of implementing a GPU-based IBM together with a set of solutions for them. Second, we hope to present an objective comparison between the cost of operating GPU-based and CPU-based IBMs and the efficiency of each solution. Finally, we hope that by presenting our model to the ocean modeling community, we may encourage others to use it in their own research. Together, we hope that this presentation will allow researchers to make more informed decisions and choose more efficient solutions regarding their own modeling efforts.

3.2 GPU Computing Overview

Although originally designed for graphics processing and rendering, GPUs have recently become a popular option for computationally intensive general purpose computing. However, general purpose computing on graphics processing units (GPGPU) presents a unique set of challenges. To facilitate better understanding of these challenges, we present an informal and simplified comparison of CPUs and GPUs here with a focus on the specific models that we used for performance testing our IBM. The details of computing technology change rapidly, and we refer the reader to the documentation provided by manufacturers (e.g. NVIDIA Corporation, Advanced Micro Devices, Inc., and Intel Corporation) for the technical details of current technology. Our presentation focuses on NVIDIA Corporation's proprietary Compute Unified Device Architecture (CUDA).

At the hardware level, CPUs and GPUs differ in their allocation of resources among subcomponents. Both CPUs and GPUs consist of control units (CUs) that translate software instructions into electrical signals, arithmetic logic units (ALUs) that perform arithmetic or logical operations based upon the signals received from the control unit, and a small amount of fast cache memory to store data.[1] However, whereas CPUs bundle a CU together with a few ALUs to create general purpose cores, GPUs bundle many ALUs with a few CUs to create specialized, parallelized cores (Figure 3-1; Couturier, 2014). Under the GPU computing model, the cache memory is shared among a much larger number of ALUs, so the amount available to each ALU is substantially lower. Additionally, the clock speed, or number of instructions processed per minute, is generally lower for GPUs than CPUs (Couturier, 2014). For comparison, the Intel Xeon E5-2650 CPU contains 8 cores running with a base clock speed of 2 GHz and 20 MB of cache memory (Intel Corportation, 2017). In contrast, the NVIDIA K20 GPU contains 2496 ALUs (CUDA cores) running at a clock speed of 706 MHz (NVIDIA Corportation, 2012). The K20 ALUs are grouped into 13 streaming multiprocessors that each contain 192 ALUs and 6 CUs. Assuming that each ALU takes a single clock cycle to perform each instruction[2], the K20 is capable of executing $1.76 \cdot 10^{12}$ instructions per second, and the E5-2650 is capable of $1.6 \cdot 10^{10}$ instructions per second. As a result, CPUs, which seek to minimize latency, can complete each instruction faster due to the higher clock speed, but GPUs, which seek to maximize bandwidth, can complete more instructions per unit time due to the larger number of ALUs.

The hardware differences between CPUs and GPUs result in different software execution patterns for each. Multi-core CPUs such as the Intel E5-2650 execute using what is known as the multiple instruction, multiple data (MIMD) pattern. Under the MIMD pattern, each processor core operates independently and asynchronously, and each core may execute a different stream of instructions on a different piece of

[1]Note that the exact terminology used differs among manufacturers. For instance, NVIDIA's CUDA cores are effectively equivalent to ALUs as defined here.

[2]This assumption is a simplification of reality. In practice, the type of data being processed and processor architecture may increase or decrease this rate. However, an in-depth discussion of the topic is beyond the scope of this study.

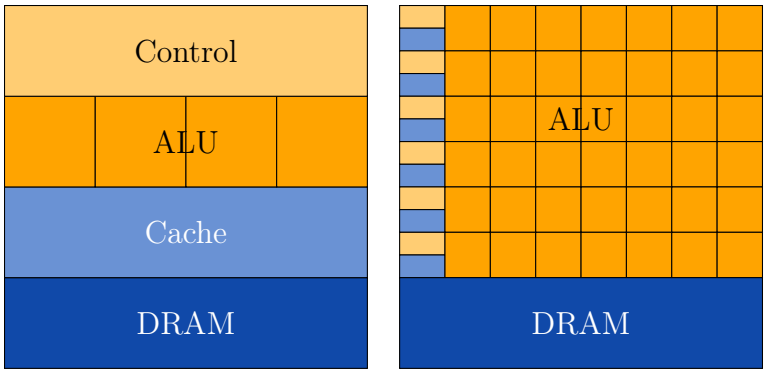

Figure 3-1: Simplified diagrams of the CPU (left) and GPU (right) computing architectures highlight the differences between them. Control units dispatch instructions to arithmetic logic units (ALUs) that perform the computations. Recently accessed variables are stored in fast cache memory, and other data is stored in slower DRAM. In practice, multiple levels of cache are used and the control flow is more complex than that depicted here. This figure was based on one that appears throughout the CPU-GPU comparison literature.

data simultaneously (Rauber and Runger, 2012). In contrast, GPUs operate using the single instruction, multiple data (SIMD) pattern, where each ALU operates on a different piece of data, but all of the ALUs within each streaming multiprocessor are required to execute the same instruction stream simultaneously (Couturier, 2014). A function containing instructions that will execute on the GPU is known as a kernel, and the CUDA framework achieves the SIMD pattern by creating threads of execution within each kernel. Each thread contains the same set of instructions, but also contains a unique identifier so that code in the thread may locate the correct data on which to operate. The CUDA runtime environment groups threads together into warps of 32 threads, and each warp is mapped onto a single streaming multiprocessor. When branching statements (e.g. `if {...} else {...}`) cause divergence between the threads in a warp, some of the ALUs on each streaming multiprocessor sit idle. As a result of this, efficient GPU code is characterized by predictable execution patterns with minimal divergence. The memory structure of GPUs generates additional implications for GPU program design. Modern processing units have multiple levels of memory that generally decrease in size and increase in speed moving up the hierarchy (Tanenbaum and Bos, 2015). The CUDA memory hierarchy as exposed to the programmer may be split into 3 levels (Sanders and Kandrot, 2010). The fastest and smallest level is thread local memory, which can only be accessed from within a single thread and is usually stored in registers. The second level in the hierarchy is user configurable as a combination of L1 cache and shared memory. L1 cache is automatically used by the CUDA runtime environment when an insufficient number of registers are available, and shared memory may be accessed by the user code to allow cooperation among threads. When the user launches each kernel, the user may specify a number of thread blocks to execute, and the number of threads within each

block.[3] All of the threads within each block will be assigned to the same multiprocessor and access the same block of shared memory. Finally, global memory is the largest, but slowest, option on the GPU and may be accessed by any thread. There are two primary ways that the memory hierarchy can be exploited to create more efficient CUDA programs. First, when many threads within each block will reuse a small amount of data, it can be loaded into shared memory to reduce the access time. Second, when many threads simultaneously request data from adjacent places in memory, the GPU device coalesces the read request into a single operation, which is substantially faster than many independent read requests.

Finally, the placement of GPUs within a computer with regards to other components requires additional computational considerations beyond CPU only code. Whereas code executing on CPUs may directly request data from hard disks and other long term storage devices at any time during the execution, code executing on the GPU can only request data that already exists on the GPU device (Sanders and Kandrot, 2010). As a result, any data that will be needed during a kernel must be preloaded into global memory prior to launching the kernel. The process of loading the data generates more complex software and consumes additional runtime.

Overall, the GPU platform provides a high performance computing platform, but is more sensitive to the software design than CPU only code. In particular, the level of branching and data locality may substantially impact performance. Ideal programs for the GPU have highly predictable execution and data access patterns that can be exploited to reduce branching and increase data locality.

3.3 Model structure

Although the principal objective of writing a GPU based IBM was to improve the computational performance, we sought to do so without compromising the ability to easily understand and modify the source code. Following the precedent established by similar models, our IBM is targeted towards Linux or similar systems and uses the NetCDF file format for data storage (Ji et al., 2012; Schlag and North, 2012; Paris et al., 2013). However, in contrast to other IBMs that are written as a collection of Fortran subroutines, the model is structured following the object-oriented programming paradigm using C++. The model may be broadly decomposed into the `FVCOMDataset` class that implements the data structures and algorithms to store and process the environment data and circulation patterns, the `ParticleGroup` collection of classes that do the same for the particles themselves, and the `IBM` class that ties them together (Figure 3-2). Our description of the model that follows describes the data structures and algorithms that were used in more detail for each of these classes. The model requires the C++ standard library and BOOST libraries for data structures, the Thrust library and CUDA framework for GPU operation, and the NetCDF C++ library for data input and output (I/O).

[3] To summarize, the assignment of threads to blocks is done by the user and is independent of the assignment of threads to warps, which is done automatically by the runtime environment.

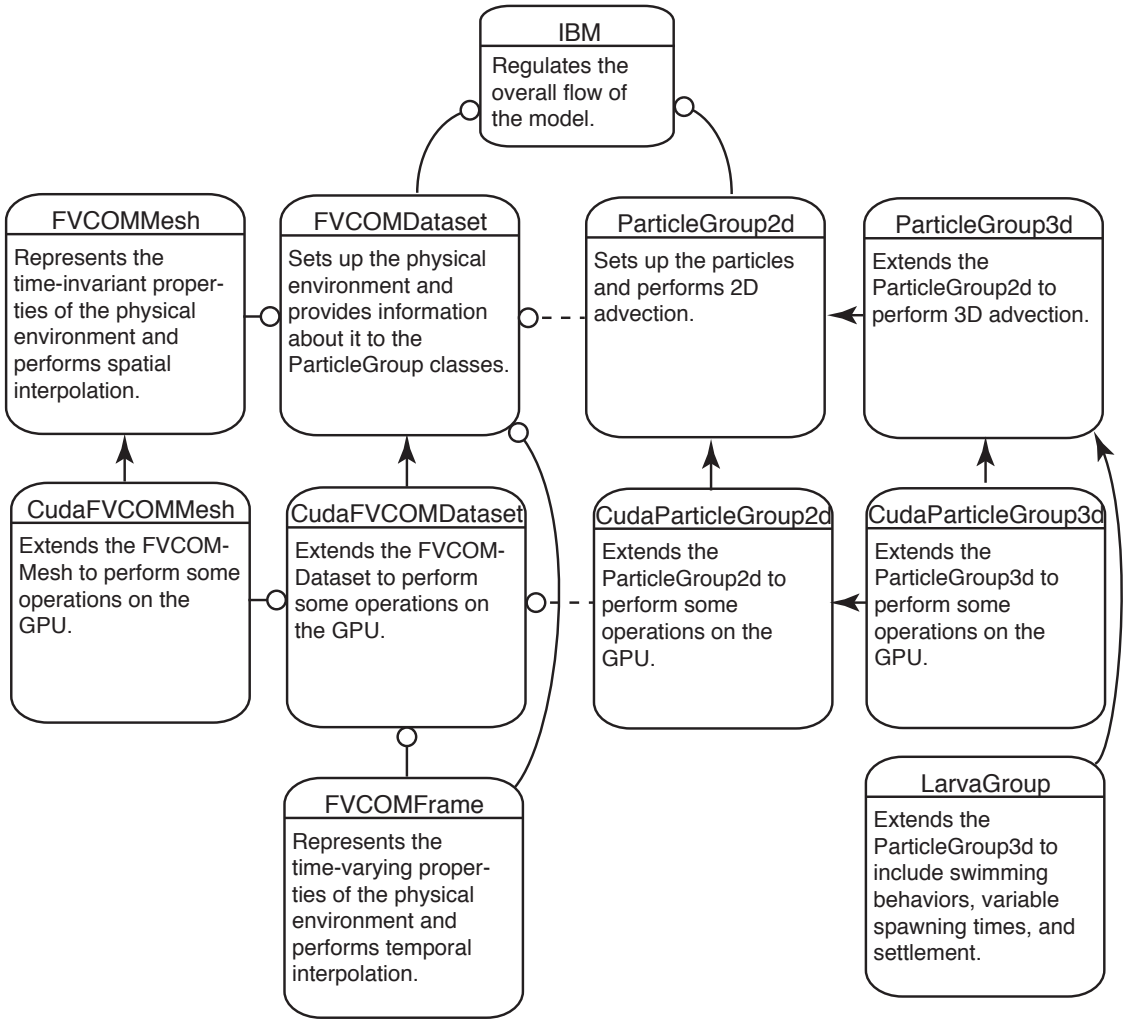

Figure 3-2: A simplified representation of the main classes composing our IBM is presented here. Arrows that terminate with a circle indicate that the origin class is a member variable in the other class. Arrows that terminate with an arrowhead indicate that the origin class inherits from the other class. Dashed lines indicate that although one class is a member of another, the member class is stored as a shared pointer and is neither created nor destroyed by the other class.

The `IBM` class has the greatest impact in regulating the overall flow of execution in the model. This class takes the name of a configuration file in JSON format as its input, and creates a set `FVCOMDataset` and `ParticleGroup` instances based on the contents of this file. From there, the `IBM` class initiates the computationally intensive main loop in which circulation fields are loaded, particle states are advanced, and the output is written to disk using a fixed, user-specified timestep, Δt. Appendix C presents an example configuration file and explains the configuration options in detail.

The objective of running our IBM is to generate a set of particle trajectories from Eulerian circulation patterns, and the `ParticleGroup` classes implement the data structures and algorithms necessary to perform this task. The particle group classes include the base `ParticleGroup2d` class and various subclasses that implement increasingly complex particle behaviors. All of the classes load the initial particle states, (x, y, z, t), from disk and advance the states in time using a modified 4^{th}-order Runge-Kutta predictor-corrector scheme. Whereas a traditional Runge-Kutta algorithm would compute the velocity vectors for the 2^{nd}-4^{th} steps using the velocity field at time $t + 0.5\Delta t$ or $t + \Delta t$, our algorithm uses the velocity field at Δt for all four steps to simplify the codebase and reduce runtime. Because the velocity decorrelation time is expected to be much longer than Δt, the approximation is unlikely to substantially influence the results. Inspecting the source code of other widely used ocean IBMs (e.g. Connectivity Modeling System, Paris et al. (2013); FISCM, Ji et al. (2012)) reveals that this approximation is standard for the field. In order to prevent particles from being advected out of the model domain, particles that otherwise would leave the model domain are returned to their previous position within the domain and continue to be advected in subsequent timesteps. The base `ParticleGroup2d` class implements advection in the horizontal dimensions only, and particles that would be advected deeper than the water depth (e.g. a deep particle moving onto a shallow shoal), track 1 m above the bottom until they return to their original release depth. A set of subclasses extend advection to 3 dimensions and provide options to specify various behaviors that are documented further in Appendix C and chapter 5.

Particle advection requires a representation of the physical environment, which is provided for our IBM by archived output from the Finite-Volume Community Ocean Model (FVCOM, Chen et al., 2006). FVCOM is a free-surface, data-assimilating model that uses the finite volume approach to numerically solve the primitive equations. The FVCOM mesh consists of a set of n triangular elements in the horizontal plane, m nodes that form the vertices of these elements, and s σ-layers in the vertical dimension. Vector quantities such as velocities are reported at each σ-layer and the center of each element, and scalar properties such as temperature or sea surface height are reported at the nodes and σ-layers where applicable. Within our IBM, the physical environment is represented by the `FVCOMDataset` class, and each particle group is provided with a pointer to a `FVCOMDataset` instance from which it may request velocity vectors, bathymetry information, etc. The `FVCOMDataset` class loads data from one or more archived FVCOM output files in NetCDF format. The time-invariant mesh data is managed using the `FVCOMMesh` class, and the time-varying environmental variables are managed using the `FVCOMFrame` class.

FVCOM archives the environmental variables at a fixed timestep that is often longer than the timestep used for particle-tracking and at fixed grid points, so a three step interpolation is necessary to obtain a value at a target point (x, y, z, t). First, the value at each FVCOM mesh element (or node) is computed by linearly interpolating between the archived values immediately preceding and following t at that element (or node). Although interpolating in time at every grid point may seem computationally inefficient, IBM runs often include many particles that reuse these values, so precomputing all of them saves time later. Second, values are interpolated horizontally at the σ-layers (or levels) immediately above and below (x, y, z). For vector quantities, this interpolation takes place using the value at the center of the element containing the point (x, y) and the values at the centers of the surrounding 3 elements. The model fits a linear plane to these 4 values, then computes the values above and below the target point. For node based quantities, the interpolation algorithm uses a linear plane fitted to the value of the quantity at the nodes that compose the element containing (x, y). The coefficients to fit the linear planes for horizontal interpolation are read from the FVCOM output when available, and otherwise precomputed to save runtime. Finally, the values immediately above and below (x, y, z, t) are linearly interpolated to the target.

The spatial interpolation uses the values from the element containing the target point, which requires locating the point within the mesh. Three algorithms are available in our model to locate the element containing each point. When the particles are loaded the beginning of each model run, the element containing each particle is unknown, and the model iterates through each element in the mesh until it finds one that contains the target point. This search operates in $O(n)$ time where n is the number of elements in the mesh, but is only necessary once at the beginning of each IBM run. During each subsequent timestep, we assume that each particle is near its location from the prior timestep. The element containing the particle at the prior timestep is known, and the first local search algorithm searches only that element and the elements that share a node with that element. The second local search algorithm follows the trajectory of the particle during the prior timestep and records each time that it crosses the boundary of an element, up to a maximum of 5 crossings. Both algorithms achieve the same goal of locating the element containing a particle, given that it has moved no more than 1 element from the prior timestep, in near constant time, and we provide additional analysis of the performance later in section 3.5. Particles that transition across multiple elements within a single timestep indicate that Δt is too large.

The model may be configured to run either entirely on the CPU or using a GPU to accelerate some of the calculations. When the model is run using the GPU, three CUDA kernels are executed. The first kernel implements the initial search to identify the element containing each particle in the mesh. The second kernel implements the time interpolation of velocity vectors and other quantities and maps each mesh element to a separate thread. The third kernel implements the Runge-Kutta integration method. Within this kernel, each particle is mapped to a separate thread within which the spatial interpolation, trajectory integration, and boundary condition checks are executed. Due to the size of the FVCOM mesh relative to the size of shared memory

on the GPU device, only thread local and global memory are used by both kernels.

3.4 Verification and Validation

Verification and validation may be informally described as insuring that the model solves the equations correctly, and that it solves the correct equations (Roache, 1998). More formally, the model verification process ensures that the algorithms described are correctly implemented, and the model validation process evaluates how well these algorithms represent the real world (Versteeg and Malalasekera, 2007). Although linked, these processes address fundamentally different goals and so we have addressed each separately.

Model verification is achieved through a comprehensive suite of unit test cases that are packaged together with the source code. These unit tests use artificially generated meshes, particle locations, forcing fields, and other parameters for which the solutions are known to check that individual methods, functions, and subroutines in our model return the correct value. For additional details about these tests, please see section 3.7 and the source code.

To validate our model, we conducted simulations using artificially generated flow fields with known solutions as well as a simulation of the Gulf of Maine using hourly archived output from FVCOM (Chen et al., 2006).

3.4.1 Flow around an obstacle validation

Our first validation test case is based on the "Flow around an obstacle" simulation described by Brickman et al. (2009) and simulates time-invariant circulation patterns. The simulation takes place on a 100 km x 50 km rectangular domain with a circular obstacle of radius R centered at the point $(x_0, 0)$. Flow far away from the obstacle moves in the positive x-direction at a speed u_0, and the streamfunction Equation 3.1 describes the velocity field.

$$\Psi = \frac{u_0 R^2 y}{(x - x_0)^2 + y^2} - u_0 y \tag{3.1}$$

For our simulation, we set $u_0 = 1 \text{ m} \cdot \text{s}^{-1}$, $R = 16$ km, $x_0 = 50$ km, and represented the domain using a mesh with 754 nodes and 1395 elements (Figure 3-3). Our model successfully reproduced the results from Brickman et al. (2009), Section 2.2.2.5 (Figure 3-3).

3.4.2 Traveling wave validation

Our second validation case describes a traveling wave that steadily translates in the x direction and represents a simple, but time-varying flow field (Samelson and Wiggins, 2006). The simulation again takes place on a 100 km x 50 km rectangular domain, but in this case there are no obstacles. The streamfunction for this flow, Equation 3.2, describes a wave with amplitude A, wavenumber k, and propagation speed c.

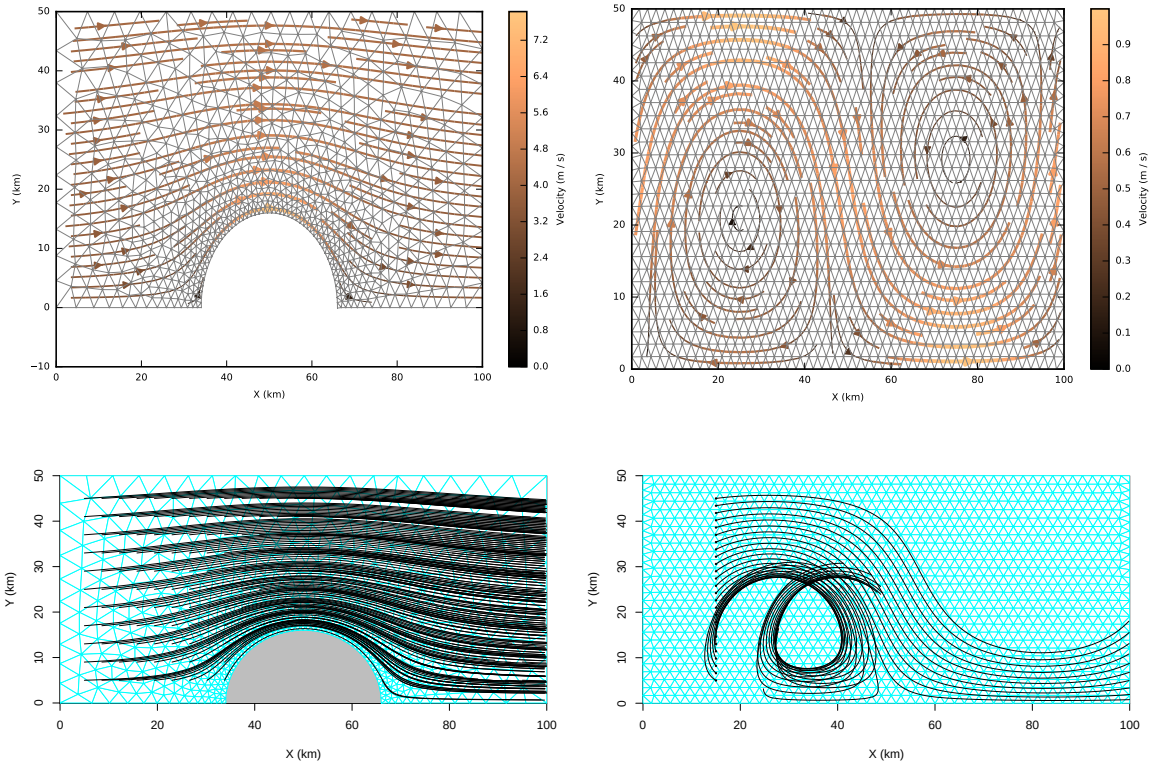

Figure 3-3: **Top left:** Streamlines for the flow around an obstacle validation case are plotted here. The black lines depict the mesh, and the velocity vectors were saved at the center of each triangular element. **Bottom left:** Trajectories for the flow around an obstacle test case as computed with our model match the streamlines. **Top right:** Streamlines for the traveling wave validation case are plotted here. The black lines depict the mesh, and the velocity vectors were saved at the center of each triangular element. **Bottom right:** Trajectories for the traveling wave validation case as computed with our model are plotted here.

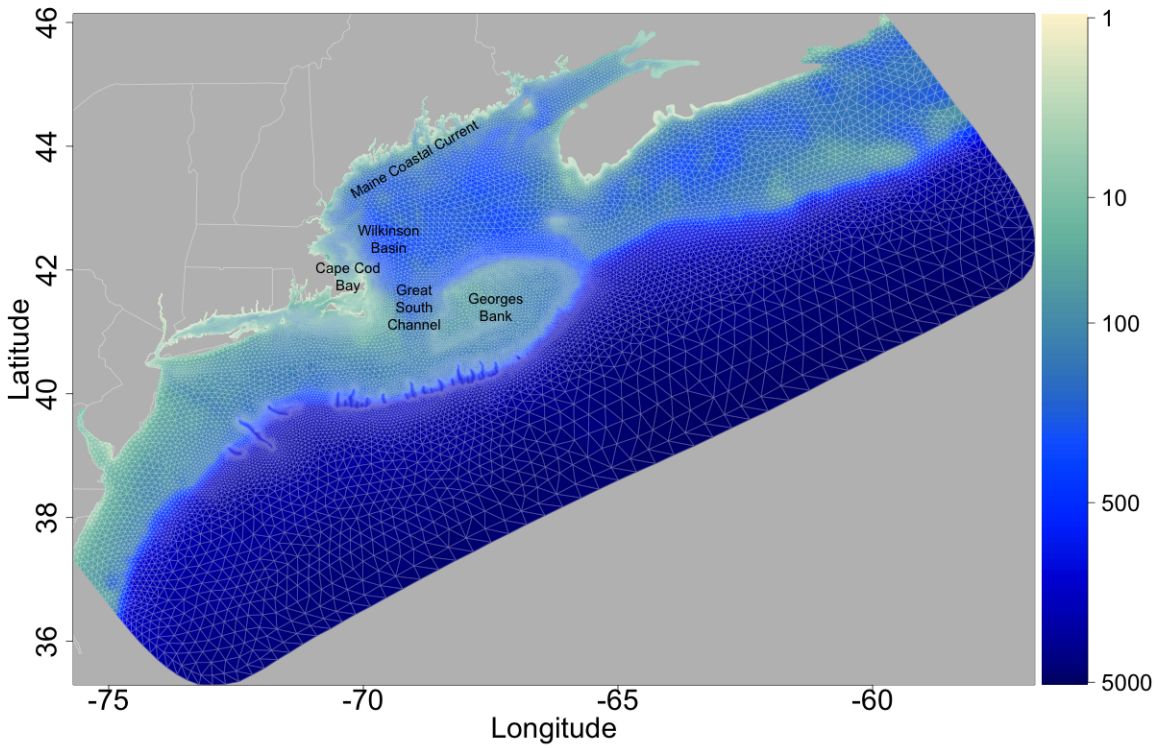

Figure 3-4: The FVCOM model domain consists of 60998 triangular elements that extend from Maryland to Cape Breton, Nova Scotia. The white lines depict the mesh elements, and the color indicates the bathymetry in meters as it is represented by FVCOM.

$$\Psi = A\sin k(X - ct)\sin Y \tag{3.2}$$

For our validation case, we set $k = 2\pi \cdot 10^{-5}$, $c = 0.5k^{-1}$, and $A = k^{-1}$ and used a mesh with 1467 nodes and 2774 elements (Figure 3-3). Trajectories that originate within the jet should follow a sinusoidal pattern, and trajectories within the recirculating regions should spiral. Our model successfully reproduced these trajectories (Figure 3-3).

3.4.3 Gulf of Maine validation

Our final validation test case used archived circulation fields from FVCOM for the Gulf of Maine and surrounding areas. Whereas the first two examples were idealized flow fields on small grids with known solution trajectories, this final test is representative of a real world use case and does not have a known solution.

The circulation fields used to force the simulation were generated by FVCOM using the 3^{rd}-generation mesh, which represents the northwest Atlantic Ocean from Maryland to Cape Breton with 60998 triangular elements, 48451 nodes, and 45 sigma layers (Figure 3-4). The mesh elements range in size from a few hundred meters

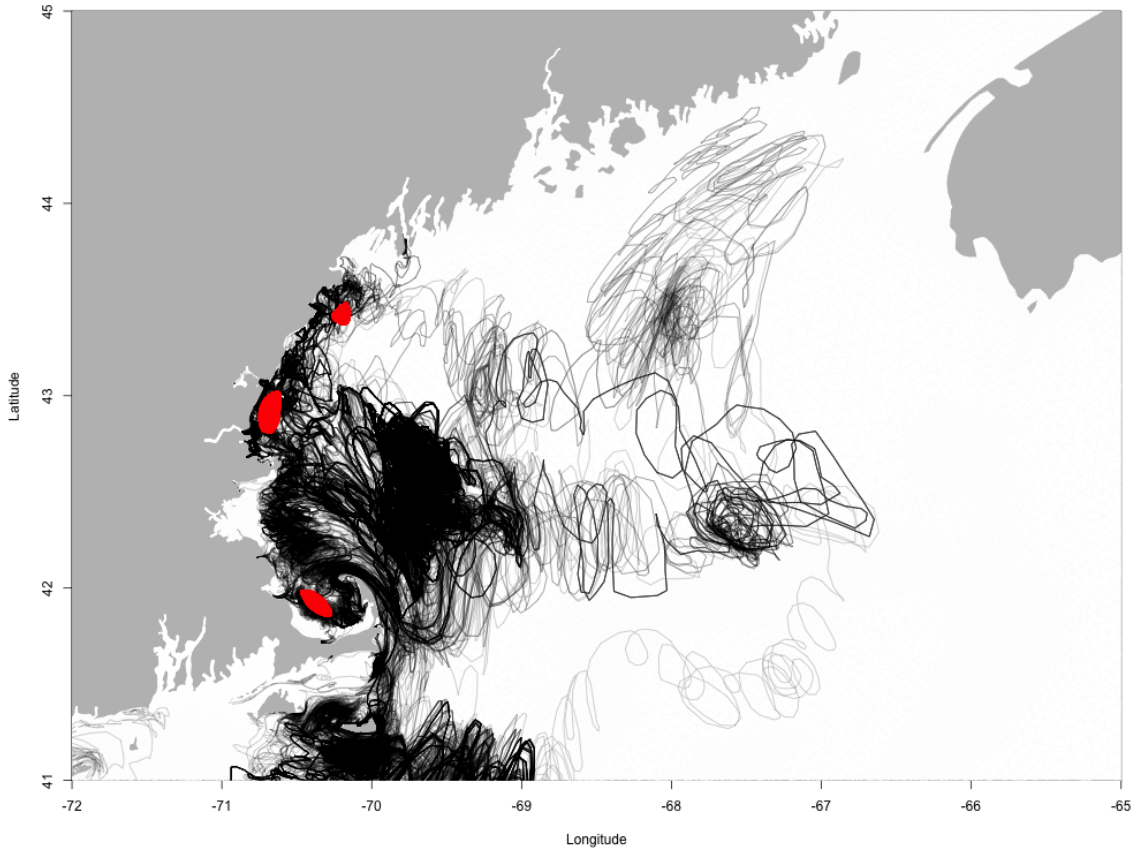

Figure 3-5: This figure shows the trajectories of 500 randomly selected particles from the Gulf of Maine validation case. The red dots indicate the release location for the particles, and the black lines are the recorded trajectories.

nearshore to 15 km in the central Gulf of Maine, and the circulation fields were archived hourly. Meshes of this size and resolution are representative of the output from state of the art circulation models for the coastal ocean.

Our particle-tracking configuration was based on the simulation of cod dispersal from chapter 2. The simulation included 150,000 passive particles that were released at midnight on 15 January 1995 at randomly chosen locations within the three spawning grounds presented in chapter 2, and each particle was advected in 2-dimensions for 60 days using a 10 minute timestep. The particle positions were recorded every 6 hours.

Overall, the particle trajectories appeared to follow the expected dispersal trajectories (Figure 3-5). Particles released from the coast of New England mostly moved south within the Maine Coastal Current. Upon reaching the coast of New Hampshire, particles either moved offshore towards Wilkinson Basin and were trapped there, or moved into Cape Cod Bay, then were swept out of the Great South Channel and south of Nantucket. Overall, these patterns replicate the expected result for the simulation.

3.5 Computational Performance

Although there are a variety of metrics available to assess computational performance of IBMs, the most important to many users is the time difference between the start and end of each run. This time difference is called the wall clock time, and we use it to quantify the performance of our model while running the Gulf of Maine validation case.

Our analysis can be split into 4 levels of detail. First, we measure the total runtime of our model running exclusively on the CPU as a serial program and compare this against the runtime of the GPU model. This test case assesses the overall performance of the model on a realistic use case. Next, we insert non-intrusive timers to time individual functions during the first 6 days of the simulation. The shorter duration of this simulation permits us to test many different configurations, and the timers provide insight into the key computational bottlenecks for IBMs. Third, we insert timers into the Runge-Kutta kernel itself to better understand which subtasks run efficiently on the GPU. Finally, we time the subroutines that identify which element in the mesh contains each point using a variety of algorithms and data structures. The results of this test help with generating hypotheses about how model performance could be further improved. Overall, the four performance testing cases range from broad scale assessment of the model as a whole to detailed optimization that highlights the specific challenges of GPU modeling.

With the exception of the element identification routines that were tested in detail, we made moderate efforts to avoid the use of costly programming language features, but did not attempt to optimize the performance of the model in depth. IBMs for ocean research are dynamic pieces of software that continually change to test new hypotheses and incorporate new processes, so detailed optimization of the full model would not be representative of their typical use patterns. We expect that additional performance gains are possible and highlight some ways that they could be achieved in section 3.6.

All of our performance testing took place on a shared supercomputing cluster equipped with Intel Xeon CPU E5-2650 CPUs running at 2 GHz and NIVIDIA K20 GPUs that contain 2496 cores running at a maximum clock rate of 706 MHz. This machine represents a typical configuration where our model would be used for research applications.

Overall, we found that the computational performance of our model is similar to other state of the art models (e.g. FISCM), and that the GPU yielded moderate speedups relative to the CPU. As a baseline, we ran a serial CPU only run. To compare the performance on the GPU, we conducted runs with a variety of kernel configurations. We report the results here for the slowest (1 thread per block) and fastest (128 threads per block) runs. Overall, the slowest GPU run was 1.4x slower than the baseline run, and the fastest GPU run was 2.1x faster than the baseline run (Table 3.1). However, much of the setup code could be parallelized on the GPU but has not been, so the results are biased towards having similar runtimes on the CPU and GPU. Considering only the timestepping routine, the slowest GPU run was 1.5x slower than the baseline run, and the fastest GPU run was 2.9x faster than the

Configuration	Total runtime	Setup runtime	Timestep runtime
CPU only	1831.9	414.0	1417.9
GPU; 1 thread per block	2495.8	411.3	2084.5
GPU; 128 threads per block	853.0	364.2	488.8

Table 3.1: The number of seconds consumed by each subprocess for the serial run on the CPU is reported here.

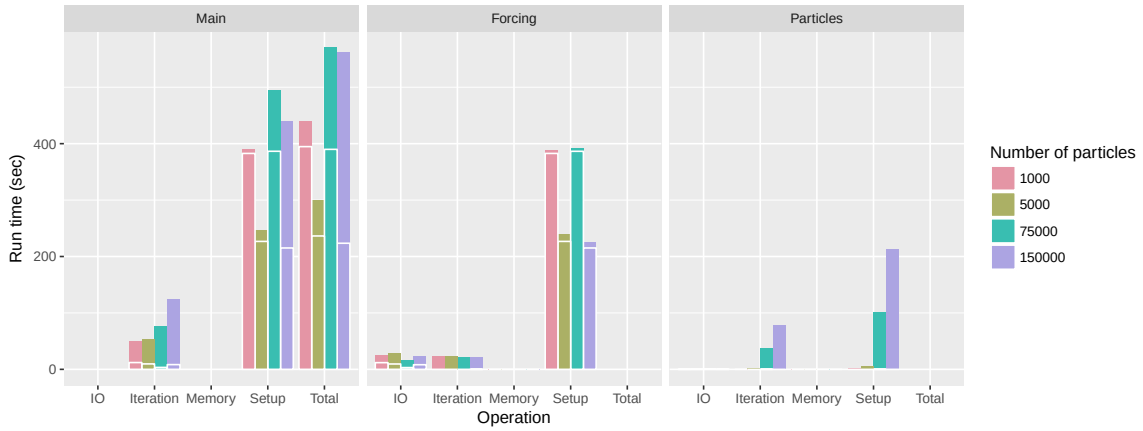

Figure 3-6: The runtime of our model on the CPU is decomposed according to the task that was being performed. The left panel shows timing results for the whole model, the center panel for tasks specific to the forcing dataset, and the right panel for tasks specific to the particle-tracking. The height of each bar is the wall clock time (the time that a user would observe using a clock external to the program) and the white outline indicates the the system time (the time spent performing memory allocations, data I/O, and other tasks that transfer control to the operating system).

baseline run.

Timers within the model revealed which tasks consumed most of the runtime. The timers themselves did not significantly increase the runtime of the model relative to equivalent runs without the timing code (Wilcoxon Rank Sum Test, n=38, V=440.00, p=0.16). Based on the CPU timers, it is possible to identify how broad categories of subtasks contribute to the model runtime. As shown in Figure 3-6, the majority of the runtime for these runs was consumed during the model setup phase. This result was expected because the number of timesteps was reduced to permit many configurations to be tested. Setting up the forcing dataset took longer than setting up the particles, and the similarity between the wall clock and system time for the forcing dataset indicates that much of the runtime here is consumed on memory allocation, data I/O, and other system tasks. In contrast, setting up the particles was dominated by user time, particularly locating the particles within the mesh. As expected, the particle setup and run times increased as the number of particles was increased, but the forcing dataset runtime was largely independent of the number of particles.

The GPU configuration with 128 threads per block took substantially less time

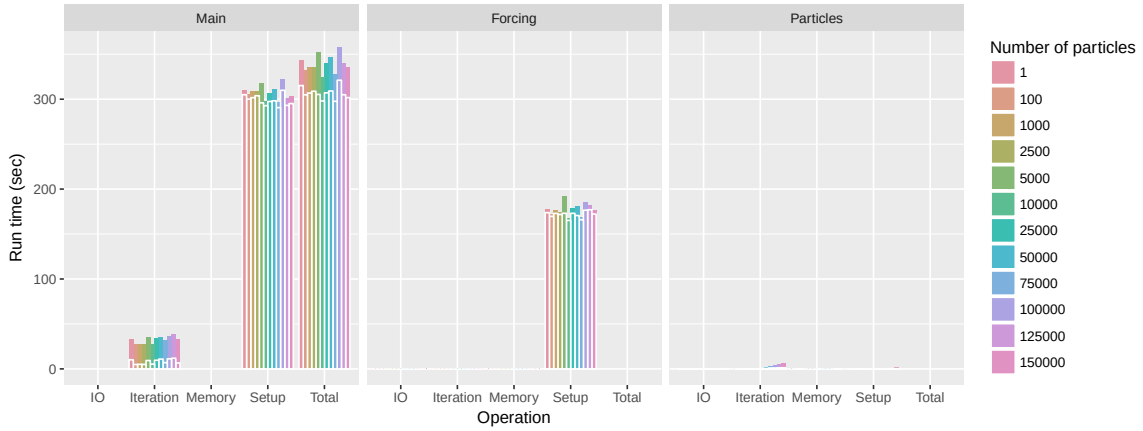

Figure 3-7: The runtime of our model on the GPU is decomposed according to the task that was being performed. The left panel shows timing results for the whole model, the center panel for tasks specific to the forcing dataset, and the right panel for tasks specific to the particle-tracking. The height of each bar is the wall clock time (the time that a user would observe using a clock external to the program) and the white outline indicates the the system time (the time spent performing memory allocations, data I/O, and other tasks that transfer control to the operating system).

than the CPU simulation, but the performance improvements were not evenly distributed throughout the model and varied based on the number of particles being simulated (Figure 3-7). Setting up the forcing dataset was not converted to run on the GPU, so the time spent on that process remained unchanged. In contrast, setting up the particles was substantially faster on the GPU. For the runs with 1000, 5000, 75000, and 150000 particles, setting up the particles was 12.15x, 77.93x, 164.10x, and 180.37x faster respectively. The bulk of the runtime for particle setup is consumed by locating the element containing each particle, and we discuss why this process is so efficient on the GPU in section 3.6. The performance increase during the timestepping on the GPU was more moderate and was 2.31x, 9.38x, 12.46x, and 13.81x faster for the 4 runs mentioned above.

To gain additional insight into how the particle runtime is allocated among the various subtasks for particle advection, we inserted timers into the Runge-Kutta kernel itself and ran the model with a variety of kernel block sizes. The timing code for GPU kernels used a different, more intrusive approach than the other timers and added 2.0 - 2.3% (interquartile range) to the runtime of the kernels. The timers highlighted that updating the particle states, which we call advection, was the least time consuming part of the process (Figure 3-8). Computing the velocity vectors, which requires interpolating in both time and space on the FVCOM mesh, took the greatest proportion of runtime. The runtime per particle increased as the number of particles increased for all of the operations, indicating that other resource limitations (e.g. memory load times), may become limitations when there are many particles.

In addition to timing the execution of the model itself, we also conducted detailed timing and optimization of one function to examine how the runtime is influenced

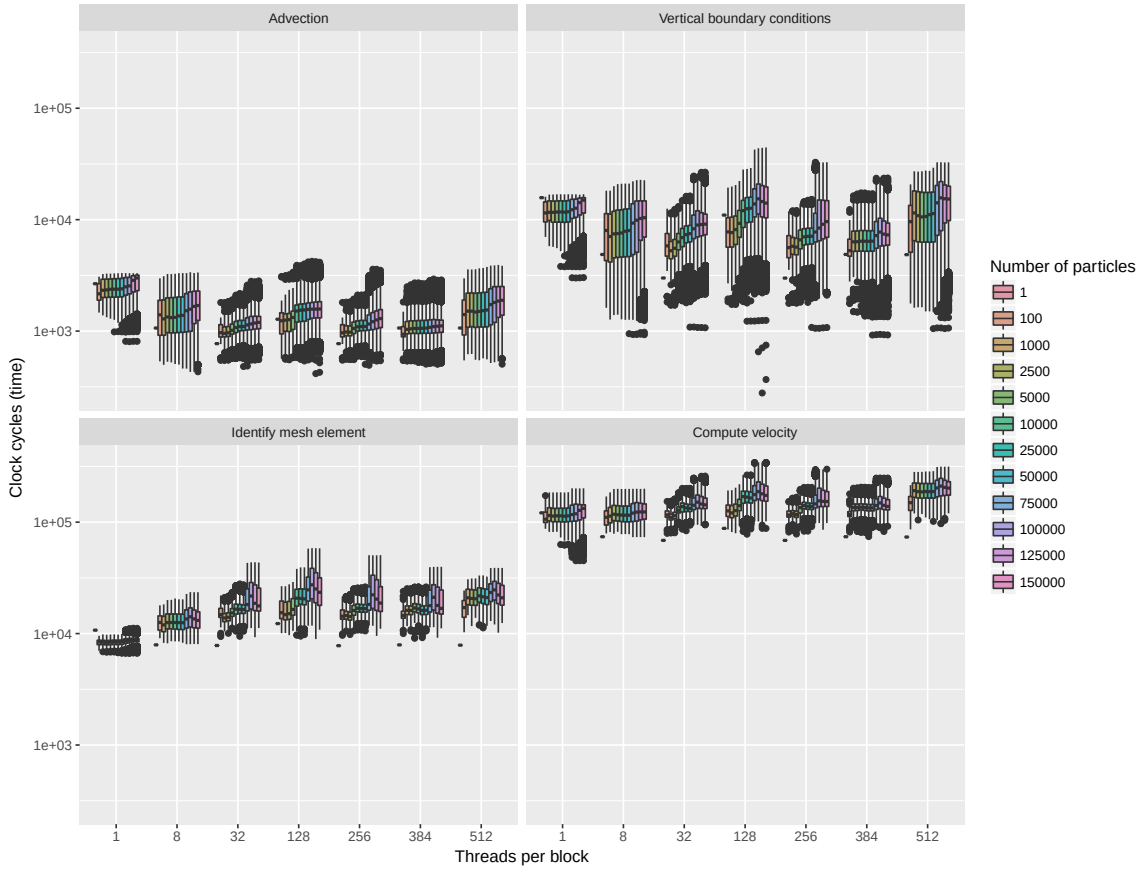

Figure 3-8: The number of clock cycles spent on each subtask per timestep of particle advection is plotted as a function of the number of threads in each CUDA block. Each data point is a single particle within the run and the color of each boxplot indicates the number of particles in the run. Each box contains 50% of the relevant data, the line in the center of each box is the median, the whiskers indicate the remaining data, and the dots are outliers.

Location	Algorithm	α	β	γ	δ	Adj-R^2
CPU	Robust	$1.1 \cdot 10^{0}$	$1.2 \cdot 10^{-3}$	$-1.6 \cdot 10^{-1}$	$-4.9 \cdot 10^{-5}$	0.86
CPU	Neighborhood	$1.5 \cdot 10^{-3}$	$3.9 \cdot 10^{-7}$	$3.8 \cdot 10^{-3}$	$8.9 \cdot 10^{-8}$	0.96
CPU	Edge	$4.2 \cdot 10^{-4}$	$1.6 \cdot 10^{-7}$	$3.6 \cdot 10^{-4}$	$1.9 \cdot 10^{-8}$	0.99
GPU	Robust	$7.4 \cdot 10^{-2}$	$6.5 \cdot 10^{-6}$	$1.0 \cdot 10^{-2}$	$7.9 \cdot 10^{-7}$	0.97
GPU	Neighborhood	$3.6 \cdot 10^{-4}$	$2.9 \cdot 10^{-8}$	$1.2 \cdot 10^{-6}$	$1.1 \cdot 10^{-9}$	0.98
GPU	Edge	$1.2 \cdot 10^{-4}$	$9.1 \cdot 10^{-9}$	$-2.3 \cdot 10^{-6}$	$6.5 \cdot 10^{-10}$	0.98

Table 3.2: The coefficients for the regression models used to predict the runtime for the search routines are presented here. Assuming that n points were located and that s is an indicator variable that takes value 1 if the SOA data type was used for the mesh and 0 if not, the model fit was time$= \alpha + \beta n + \gamma s + \delta ns$. Coefficients are reported as the estimate $\pm$ standard error.

by the number of particles, the choice of algorithm, and the data structures. The function we tested identifies the element within the FVCOM mesh that contains each point. We generated 100000 points within the Gulf of Maine mesh and searched for varying number of these points using the robust search, the neighborhood search, and the edge search algorithm. In each case, we tested the search routines both using an array of structures (AOS) storage type and a structure of arrays (SOA) data storage type. The choice to use AOS vs. SOA data structures influence how the coordinates of the nodes and the indices of the element vertices are stored in memory. Whereas the AOS data type interlaces the x and y coordinates of the nodes and the vertex indices for the element in memory (e.g. $x_0, y_0, x_1, y_1, ...$), the SOA data type would result in all of the x-coordinates being stored adjacent to one another, followed by the y-coordinates, then the first vertex of each element, and so on (e.g. $x_0, x_1, x_2, ..., y_0, y_1, ...$). We ran 3 replicate simulations for each combination data structure type, number of points being searched for, run location (CPU or GPU), and search algorithm. After separating the runs based on the run location and search algorithm, we fit linear regressions that model the wall clock time as a function of the other terms plus the interaction between them. The coefficients for these models are reported in Table 3.2. Overall, the robust search algorithm was the slowest on both the CPU and GPU, and the edge search was the fastest. The robust search took $1.2 \cdot 10^{-3}$ seconds per particle on the CPU, and was 3 order of magnitude faster on the GPU at $6.5 \cdot 10^{-6}$ seconds per particle. The neighborhood and edge searches were 4 orders of magnitude faster than the robust search on the CPU, but only 2 orders of magnitude faster on the GPU, indicating that the GPU is more effective at accelerating the robust than local search. The effect of the AOS vs. SOA data structure was 1 to 2 orders of magnitude less than the overall runtime in all cases, indicating that it has a small, but potentially meaningful effect on the runtime.

Overall, the GPU offered modest acceleration relative to a CPU-only run. The performance improvements were most extreme for the robust search that takes place during the setup phase, and less meaningful for the local search and interpolation processes that happen each timestep.

3.6 Discussion

This study presents evidence that GPU-based modeling has the potential to improve the performance of IBMs, but that the benefits are not evenly distributed throughout the software. Our results suggest that IBMs which distribute the workload across both the CPU and GPU may most efficiently make use of computing resources.

Overall, the performance improvements that we observed by transitioning subtasks to the GPU were similar to other studies. Herault et al. (2010) found that executing their smooth particle hydrodynamics model on a GPU resulted in speedups ranging from 3.2x for some subtasks to 207x for others. Engsig-Karup et al. (2014) tested a non-linear, dispersive free surface water wave model on a heterogeneous system that included multiple GPUs and observed speedups of up to 2 orders of magnitude depending on the model configuration and subtask being considered. Lee et al. (2010) presents the efforts of a group of engineers from the CPU manufacturer Intel to compare CPU and GPU performance after carefully optimizing the code for both, including parallelizing the CPU code to fully exploit its computational ability. They observed an average speedup of 2.5x across a range of possible applications, and the performance ranged from 2x slower to 15x faster on the GPU.

One of the key findings from this study and from prior ones (e.g. Lee et al., 2010; Herault et al., 2010; Engsig-Karup et al., 2014) is that the performance improvement from executing code on the GPU is highly dependent on the application. In the case of our IBM, tasks with a high compute intensity to memory access ratio, such as the robust element search, were up to 3 orders of magnitude faster on the GPU than CPU. However, other tasks, such as the local search algorithms, were accelerated substantially less by the GPU. This result emerges because GPUs dedicate resources to ALUs that perform calculations. Accordingly, they have smaller memory caches and less memory bandwidth per core than CPUs. To make efficent use of this architecture, software must perform many computations on each piece of data that is loaded. For example, our robust search algorithm loads the coordinates of each element, then uses these coordinates to simultaneously check if many particles are within that element. In contrast, the neighborhood and edge based search routines search a different element for each particle under consideration, and so the coordinates of many elements must be read. Expending effort to parallelize compute intensive tasks such as the robust search is likely to result in substantial performance improvements for existing IBMs with minimal effort.

The size of the forcing datasets used by oceanographic IBMs also limits the potential for GPU-based modeling. Many IBMs rely on high resolution circulation fields to represent the physical environment. The circulation patterns can be most accurately and efficiently calculated when the hydrodynamic model mesh tracks bathymetric or density gradients. Unfortunately, these meshes also require that the coordinates of each node and element be loaded into memory before accurate interpolation is possible, and the meshes are often too large to fit into the fast shared memory banks on GPUs. Together with the largely random access pattern that is dependent on the location of each particle in the mesh, local search and interpolation routines on GPUs may result in many independent and inefficient read operations from global memory.

Both of the above mentioned limitations on GPU-based IBM performance may be partially alleviated through changes to the software. In the current version of our model, the mesh nodes and elements are stored in the SOA format and are organized to minimize the amount of memory used. That is, the nodes are stored as three arrays that contain the x, y, and h coordinates for each node, and the mesh elements are stored as arrays containing the index of the first, second, and third node bounding each element. As a result of this storage format, loading an element requires first reading the the indices of the three nodes that bound it, then reading the x-coordinate and y-coordinate of each nodes. The results of the linear regressions presented in Table 3.2 suggest that switching to the AOS format may reduce the runtime of the search routines on the GPU. Extending this change further would involve creating a new array to store the coordinates of all three nodes that border each element adjacent to one another in memory. Although this change would increase the memory footprint of the IBM, it would also reduce the bandwidth required to load the elements and may decrease the runtime.

A more complex option to improve performance would be to decompose the mesh into a set of smaller submeshes, each of which is small enough to fit in shared memory. Particles may then be tracked as they transition among submeshes, and the GPU kernels could be designed so that each kernel block processes only particles residing on a specific submesh. Although this approach could potentially improve the GPU-based IBM performance substantially, it would add a nearly unmanageable amount of complexity to the code.

The most promising approach is most likely to run IBMs on both CPUs and GPUs. Certain tasks that run efficiently on the GPU, such as the initial search to identify the element containing each particle, could be run there. Other tasks benefit less from GPU acceleration and could continue to be executed on the CPU. This approach would yield much of the performance benefit of GPU-based modeling and would result in substantially fewer modifications to existing IBMs than running the entire model on the GPU. Overall, our results suggest that GPUs may be a useful tool to improve IBM performance, but are unlikely to completely displace CPUs as the primary architecture in the near future.

3.7 Code availability

The source code for our model is available at `https://github.mit.edu/btjones/nemo`. Installation instructions are also available as part of that repository.

Chapter 4

Identifying coherent geographic regions from population connectivity data

Abstract

Mathematical graphs are abstractions that represent discrete objects and the relationships among those objects. Within marine ecology, graphs may be used to represent the probability of larval transport among discrete geographic regions in the ocean, and graph theory may be used to analyze the transport patterns. The development and analysis of these graphs has recently been used in oceanography to identify important populations for conservation and to describe hydrodynamic provinces from population connectivity data. A fundamental component of this analysis is choosing appropriate boundaries between the geographic regions, and this discretization process may be influenced by preconceived opinions of the researchers. One method to address this issue is to initially discretize the study area into smaller regions than necessary, then cluster these small regions together based on the connections between them. The resulting larger, coherent regions can then be used for later analyses. This study compares two different algorithms that have been separately used for marine ecology studies and presents a techinque for generating equivalent clusters from the different algorithms. It also extends one of the existing algorithms to consider multiple species when choosing the clusters and demonstrates the operation of this new algorithm using 4 species that live in the Gulf of Maine. Overall, we find that the existing algorithms identify similar clustering patterns, but the ease of relating their parameters to ecology varies between methods. The clusters resemble published stock structure patterns, and our new method supports hypothesized relationships between the species.

4.1 Introduction

Marine population connectivity, or the movement of individuals among geographically separated subpopulations, is a fundamental process that governs ecosystem structure (Cowen and Sponaugle, 2009). Often, it is studied by discretizing the study region into a set of discrete geographic regions, then estimating the probability that each individual will transition from one region to another. Using this approach, it has been possible to highlight potential avenues of species expansion (Treml et al., 2008), identify important sites for conservation efforts (Watson et al., 2011), and hypothesize about potential impacts of contamination events on population connectivity (Jones et al., 2015). One of the core requirements for this approach to be successful is that the geographic discretization process yields meaningful regions. This study focuses on this process and examines multiple algorithms that have been used in marine ecology to identify coherent geographic clusters from population connectivity data. We seek to elucidate and reconcile the differences among different clustering algorithms that have been used, demonstrate the capabilities of existing clustering algorithms, and describe a new algorithm that permits automated inter-specific comparisons of transport patterns.

Graph theory is the study of mathematical graphs that encode the relationships among a set of discrete objects and has emerged as a promising technique for analyzing population connectivity data (Baranyi et al., 2011). Each graph is composed of a set of nodes that represent the objects and edges that represent pairwise connections among the nodes. When applied to population connectivity, each graph node represents one of the origin or destination sites, and the edges represent the probability that a larva released from a particular origin will settle at a particular destination (Thomas et al., 2014). In this case, the edges are directed so that the connection from a to b may differ from the one from b to a, and the graph may include self-links that begin and end at the same node. Graphs may encode the same information as and are interchangeable with the connectivity matrices discussed in chapter 2.

A number of methods have been developed to extract ecologically relevant information from connectivity matrices and graphs at varying levels of detail. Graph level metrics such as the clustering coefficient, diameter, and self-recruitment rate summarize properties of the graph as a whole (Baranyi et al., 2011). These metrics provide insight into the the graph structure, scales of dispersal, and openness of the population. Other metrics, such as the betweenness and eigenvalue centrality, highlight individual nodes that are particularly important for maintaining connectivity or sustaining the population. Many of the graph and node level metrics are redundant, and Baranyi et al. (2011) describes how a subset of them may be used to summarize many of the graph properties. Although computing an appropriate subset of the available summary metrics can create a concise description of the important population connectivity patterns within a graph, this summary is dependent on the assumptions that went into constructing the graph. The nodes, which represent geographic regions, are particularly influenced by researchers' opinions and are generally chosen based on preconceptions of the oceanographic and ecological structure. For example, splitting a particularly important spawning site into multiple nodes or merging it with a less

important spawning area will downplay the importance of that site. Unfortunately, however, understanding the properties of the nodes, not the edges, is the central focus of population connectivity studies because resource managers may regulate the usage of geographic areas, but the transport among them is largely driven by uncontrollable ocean circulation patterns and species traits. Before computing the summary metrics, it is therefore useful to ensure that the nodes have been appropriately specified.

One objective way to discretize the study domain is to compute coherent geographic regions from connectivity patterns. Efforts to identify geographic regions that are internally well connected but largely separated from one another may be broadly split into two categories. The first of these, manifold-based methods, seek to identify the boundaries that exist between regions (Ser-Giacomi et al., 2015). Lagrangian coherent structures (LCS) are manifolds that move with the flow and divide the flow into dynamically consistent regions (Haller, 2015). LCS are often computed from the deformation and stretching of a unit cube of fluid, and may be quantitatively described using a mathematical quantity known as the finite-time Lyapunov exponent (FTLE, Shadden et al., 2005). The flux across sharply defined and well-behaved positive forward-time FTLE surfaces is negligible, and particles released on opposite sides of the surface may be expected to move apart over time (Shadden et al., 2005; Harrison and Glatzmaier, 2012). Harrison et al. (2013) used an LCS-based analysis to evaluate larval transport patterns in the California Current System. They found that FTLE ridges are associated with filamentation and eddy-eddy interactions, and that these features of the circulation patterns aggregate larval into dense packets. The larval density within the packets may be up to 2 orders of magnitude greater than release densities and are robust to larval behavior, suggesting that FTLE fields may provide valuable insight into population connectivity patterns. Unfortunately, as we explain further in Appendix D, computing meaningful FTLE fields can be difficult for variable resolution circulation models.

The second type of method for identifying coherent geographic regions, set-based methods, focuses on identifying the geographic regions themselves instead of the boundaries between them (Ser-Giacomi et al., 2015). Fortunately, this topic, identifying coherent clusters from mathematical graphs, has recently been the subject of great interest from multiple fields. Recent applications of clustering procedures to mathematical graphs include diverse topics such as identifying clusters from telephone records, examining political alliances based on politicians voting patterns, and analysis of gene regulatory networks in biology (Fortunato, 2010). Within oceanography, clustering algorithms have recently been developed and applied to estimate circulation patterns from observations of scalar variables (Molkenthin et al., 2016), to describe bioregions from multiple species distributional patterns (Edler et al., 2015), and to identify hydrodynamic provinces from population connectivity patterns (Jacobi et al., 2012; Thomas et al., 2014; Rossi et al., 2014; Ser-Giacomi et al., 2015).

Beginning with a large number of small geographic cells, set-based methods seek to cluster the cells together into larger coherent regions. Multiple set-based algorithms are available, and each algorithm seeks to identify a clustering of graph nodes that minimizes a predefined objective function (Rosvall and Bergstrom, 2008; Traag et al., 2011; Schaub et al., 2012). The objective function quantifies the quality of the

clustering, and some that have been used include modularity (Blondel et al., 2008) and the constant Potts model (CPM) that is discussed later in subsection 4.2.2 (Traag et al., 2011; Thomas et al., 2014). Although the details of the objective function differ among methods, most define a good clustering as one where the density of edges that originate and terminate in the same cluster is much higher than the density of edges that cross between clusters (Rosvall and Bergstrom, 2008; Traag et al., 2011; Jacobi et al., 2012). The objective functions also often include a tunable parameter that regulates the size of the detected clusters (Jacobi et al., 2012; Thomas et al., 2014). In subsection 4.2.1 and subsection 4.2.2, we present further details about two such algorithms and objective functions that have been used for marine ecology research. Jacobi et al. (2012) optimized the CPM to detect coherent geographic regions from the results of an individual-based model (IBM) simulation for cod in the Baltic Sea. By sweeping across a range of values for the tuning parameter, they constructed a suite of possible clusterings with varying numbers of clusters but with similar compositions. In doing so, they found that the marine protected areas (MPAs) in the area were not evenly distributed across the clusters, and so some subpopulations may be better protected than others. Thomas et al. (2014) applied the CPM to identify clusters from simulated data for multiple species in the Great Barrier Reef. They again compared MPA placement against larval dispersal information, and although they found that the MPA placement is generally higher in nearshore areas with shorter dispersal distances, they did not directly compare the MPA placement against the identified clusters. Other studies have used a different algorithm called Infomap to study connectivity patterns in the Mediterranean Sea (Rossi et al., 2014; Ser-Giacomi et al., 2015). Rossi et al. (2014) used the Infomap program to evaluate the structure of an MPA network in the Mediterranean Sea, and Ser-Giacomi et al. (2015) built upon their results to compare Infomap against other methods for characterizing geophysical flow transport. Rossi et al. (2014) found that the clusters identified by Infomap are highly coherent and match the mean streamlines of the flow well, but that the MPAs are not evenly distributed across the clusters. Ser-Giacomi et al. (2015) found that the Infomap clusters were different from those that would result from an older set-based method, spectral partitioning and that the Infomap clusters were less uniform in size and contained greater detail. Overall, the set-based methods appear to perform well at detecting ecological clusters.

Both the manifold-based and set-based algorithms have the potential to be highly useful for ecological studies, but some enhancements would greatly increase their utility. Ser-Giacomi et al. (2015) compare the results of LCS, Infomap, and spectral partitioning, and although they conclude that Infomap performs well, they also note that the objective function and tuning parameter do not have a clear ecological meaning. We seek to build upon their results by clarifying the relationship between Infomap and ecology and by comparing the results of the Infomap and CPM-based approaches. Finally, these approaches are limited to a single connectivity matrix, but the marine environment is a dynamic and constantly changing environment. Even with a common set of ocean circulation patterns, the transport patterns between species differ. We conclude with a new algorithm that considers the dynamic environment and inter-specific differences to simultaneously cluster nodes geographically

into coherent clusters and among species according to the similarity in their transport patterns.

The objectives of this study are to clarify the ecological meaning of Infomap and CPM-based methods for identifying coherent clusters in graphs, to reconcile the differences among these methods, explaining when and why they may be expected to arrive at different results, and to present and demonstrate a new multigraph method that can be used to simultaneously compare graphs while clustering the nodes within each.

4.2 Clustering Algorithms

Consider a study system that consists of m discrete geographic regions, and let p_{ij} be the probability of a larva that was spawned in region i settling in region j. In chapter 2, we represented this study system using a connectivity matrix, $P = [p_{ij}]$, and developed a method to estimate P from IBM output. In this chapter, we represent the system using an alternative abstraction known as a mathematical graph. Applied to population connectivity data, each node within the graph represents a geographic region, and each edge represents a transport probability, p_{ij}. Mathematical graphs that represent population connectivity data are thus interchangeable with and contain equivalent information to connectivity matrices. Each of the algorithms that we describe next seek to cluster nodes together such that the nodes within each cluster are strongly connected to one another and only weakly connected to nodes in other clusters. Although Infomap, the modified Louvain method, and the multigraph method all use an optimization heuristic to minimize the value of an objective function that quantifies the clustering quality, the details of the optimization heuristic and objective function differ among them.

4.2.1 Infomap

The first algorithm that we describe, Infomap, seeks to form coherent clusters based on the multi-generational flow of larvae through the geographic regions (Rosvall and Bergstrom, 2008). We provide only a brief summary of Infomap here and focus on its ecological relevance, but the full details of the algorithm are available from Rosvall and Bergstrom (2008). Consider an individual that originates from a randomly chosen node within the graph. Each generation, this individual produces a single offspring that transitions to one of the other nodes based upon the transport probabilities, and the original individual passes away. Over the course of many generations, the descendents of the original individual follow a random path through the network that is influenced by the transport probabilities. Infomap simulates this process for a large number of individuals and attempts to concisely encode the paths that are followed by each lineage. It has been long known that a Huffman code, where frequently visited nodes are assigned short names and rarely visited nodes assigned longer names, provides a concise encoding of the path. The novel feature of Infomap is that instead of using globally unique identifiers for each node, Infomap uses identifiers that are unique

within each cluster together with codewords that indicate a transition among clusters to shorten the encoding. The size and number of identified clusters is regulated by the Markov time, which is the number of times each individual transisitions among nodes before including its position in the path followed by that individual. For example, the default Markov time of 1 results in every node visited by the individual being included in the path, a Markov time of 2 would result in every other node being included, and a Markov time of 0.5 would result in every node being included twice. Infomap then chooses the clustering that results in the fewest number of bits required to encode the trajectories through the graph. This clustering may include multiple nested levels of clusters. Rosvall and Bergstrom (2008) provide additional details about how the clusterings are created and refined, but those details are not relevant to its ecological application.

4.2.2 Modified Louvain Method

The second algorithm that we describe uses the CPM as an objective function and the Louvain method as an optimization scheme (Blondel et al., 2008; Traag et al., 2011; Thomas et al., 2014). The CPM expresses a tradeoff between the density of edges within each cluster and the size of the clusters. Its value is given by Equation 4.1, where p_{cc} are the transport probabilities that originate and terminate in cluster c, m_c is the number of nodes in cluster c, and γ is a tuning parameter that regulates the size and number of clusters. Under the optimal clustering, the mean connection strength between nodes within each cluster will be at least γ, and the mean connection strength between nodes in different clusters will be no more than γ (Thomas et al., 2014).

$$\mathcal{H} = -\sum_{c} p_{cc} - \gamma m_c^2 \tag{4.1}$$

Although a variety of techniques could be used to optimize the value of $\mathcal{H}$, Thomas et al. (2014) use the simple but effective Louvain method for their marine population connectivity study. The Louvain method consists of two phases that are repeated and was originally designed to optimize a different metric called modularity (Blondel et al., 2008). Initially, each node is assigned to a unique cluster. The first phase of the algorithm consists of iterating through the nodes and moving the nodes among clusters to minimize the value of $\mathcal{H}$. Let c_i be the cluster that contains node i, and consider the change that would result in the objective function from moving this node to the cluster containing node j, c_j. If the change in the objective function is negative from this change, then setting c_i to c_j would improve the value of the objective function. The Louvain method iteratively considers this movement for every node i and cluster c_j, provided that the connection between nodes i and j, p_{ij}, is positive. Once the algorithm completes one full pass through all of the nodes without any movements, it then moves onto the second phase. During the second phase, all of the nodes within each cluster are merged together into a single node. All of the edges that originate from a node within cluster i and terminate at a node in cluster j are summed together and used to create a single edge that connects the clusters. The

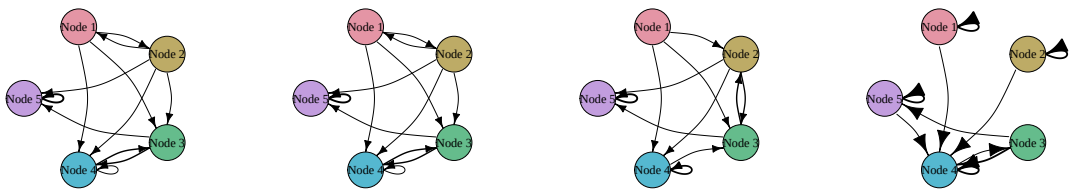

Figure 4-1: Four graphs, each containing the same 5 nodes, are used to demonstrate the multigraph method. The width of each arrow indicates the strength of each edge, and the color of each node indicates the cluster to which it belongs. Each node here has been assigned to a unique cluster.

first phase is then repeated on the reduced size graph to generate a nested hierarchy of clusters. Because the Louvain method as used by Thomas et al. (2014) and in this paper is used to optimize the CPM instead of modularity, we refer to it as the modified Louvain method.

4.2.3 Multigraph Method

Whereas the Infomap framework and Louvain method cluster the nodes within a single graph, the multigraph method operates on multiple graphs simultaneously. In addition to clustering the nodes within each graph, the multigraph method clusters the graphs together into regimes based on their similarity to one another. As with the other methods, the multigraph method is a hierarchical clustering algorithm that results in a dendrogram. Previously, each node in the resulting dendrogram represented a cluster of graph nodes and contained only the indices of the graph nodes in that cluster. Under the multigraph method, each dendrogram node represents both a cluster of graph nodes and a regime of graphs that are similar to one another.

Assume that the study system consists of n graphs, $G = \{g_1, g_2, ..., g_n\}$, and that each graph contains m nodes. Further, let n_{ij} be node j in graph i, and assume that $n_{ij} = n_{kj}$ for any i and k. That is, the geographic regions that correspond to each node are the same across all of the graphs, and only the edge weights differ between graphs. To make the description of this method more concrete, we rely on a visual example using 4 graphs with 5 nodes each. Initially each graph node is assigned to a unique cluster within that graph, and each graph is assigned to a unique regime (Figure 4-1).

The multigraph method is an iterative algorithm that involves repeating a sequence of three steps until the entire sequence is completed without any further changes to the results. Each iteration begins by executing the first phase of the modified Louvain method on each graph. As a reminder, this process involves clustering the nodes together within each graph to minimize the value of the CPM, $\mathcal{H}$. The value of γ is chosen separately for each graph to generate equivalent clustering patterns as described in subsection 4.3.2. At the end of this step, each graph contains a set of clusters, and the member nodes within each cluster may differ between

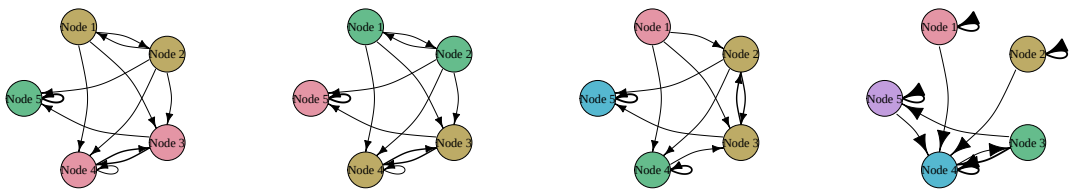

Figure 4-2: The graphs used to demonstrate the multigraph method are replotted here after completing the first step of the multigraph method. The width of each arrow indicates the strength of each edge, and the color of each node indicates the cluster to which it belongs.

graphs (Figure 4-2).

The second step of the multigraph method seeks to cluster the graphs into regimes, where each regime is a set of graphs that have similar population connectivity patterns. This step occurs in two substeps. First, a distance matrix, $D = [d_{ik}]$, is computed where each element of this matrix, d_{ik}, gives a quantitative estimate of the difference between graphs g_i and g_k. To compute this matrix, we rely on the condition that each graph in G was required to have the same nodes. As a result, it is possible to assign the clustering pattern from g_k to the nodes in g_i (Figure 4-3). Because the clustering pattern for g_i was chosen to minimize the value of $\mathcal{H}$, this projection of the clustering pattern from g_k onto g_i will increase the value of $\mathcal{H}$ for g_i. We set d_{ik} to the difference between the new value of $\mathcal{H}$ and the original value of $\mathcal{H}$, then normalize the values in D to span the range $[0, 1]$. This normalization assists in setting a tuning parameter for the second substep where D is converted to a mathematical graph and the first phase of the Louvain method is run once on this new graph. As a result of this procedure, the graphs themselves are clustered together into regimes. In our example, we would find that graphs 1, 2, and 3 all belong to regime 1 and that graph 4 belongs to regime 2.

The third step of the multigraph method merges all of the graphs within each regime into a single graph. Ideally, this merged graph would preserve both the clustering patterns and edge weights of the member graphs. To meet these ideals, the graph merge step is broken into three substeps where the first two substeps merge the clustering patterns and the third substep merges the edge weights. It is possible that graphs within a regime with identical clustering patterns will use different cluster ids for each node (e.g. graphs 1 and 2 in Figure 4-2), and the first substep attempts to rectify this issue. The graph within each regime with the most unique clusters is treated as a reference, and the cluster ids within this graph are held constant. The cluster ids for each other graph within the regime are renumbered to minimize the pairwise mismatch rate between nodes in the reference graph and each other graph (Figure 4-4). In the second substep, the member graphs of each regime are merged into a single graph. The nodes in this graph are the same as for each member

Graph 1 Graph 3 Graph 1 with Graph 3 clusters

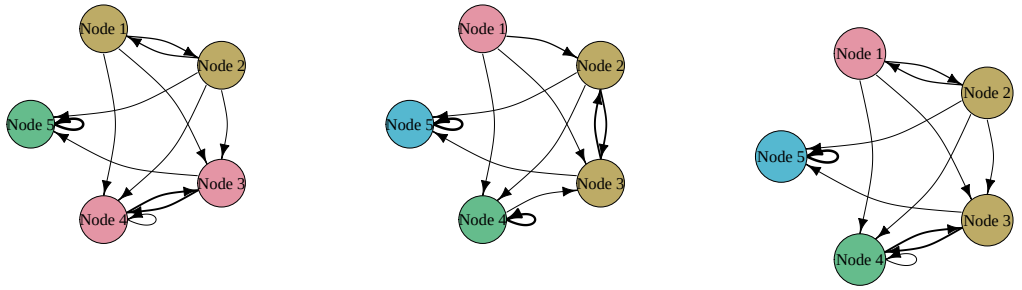

Figure 4-3: The edge weights for the regime clustering graph are computed by projecting the clustering from each graph onto each other graph. The graph on the right is formed by projecting the clusters from graph 3 onto graph 1 and would be used to compute d_{13}.

Graph 1 Graph 2 Graph 3 Merged Graph

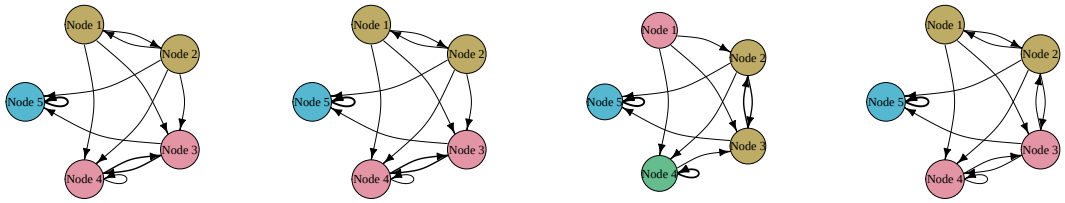

Figure 4-4: In the left 3 plots, the clusters within graphs 1 and 2 have been recolored so that they match the cluster colors from graph 3 as closely as possible. The colors for each of these graphs were determined during the first step of the multigraph algorithm. In the far right plot, the graphs have been merged into a regime averaged graph.

of the regime, the cluster id for each node is chosen by a voting procedure by the regime members, and each edge weight, p_{ij}, is the mean of the corresponding p_{ij}s from the member graphs (Figure 4-4). The third substep is equivalent to the second phase of the Louvain method. In this substep, the nodes within each cluster are merged into a single node and the edge weights are summed across these nodes (Figure 4-5).

The three steps are then repeated on the newly formed graphs. After each iteration, there are as many or fewer regimes than at the start of the iteration, and each graph has as many or fewer clusters than at the start of the iteration. The procedure terminates when there is only a single regime or when an iteration completes with the same number of regimes as it started with.

Original Graph Agglomerated Graph

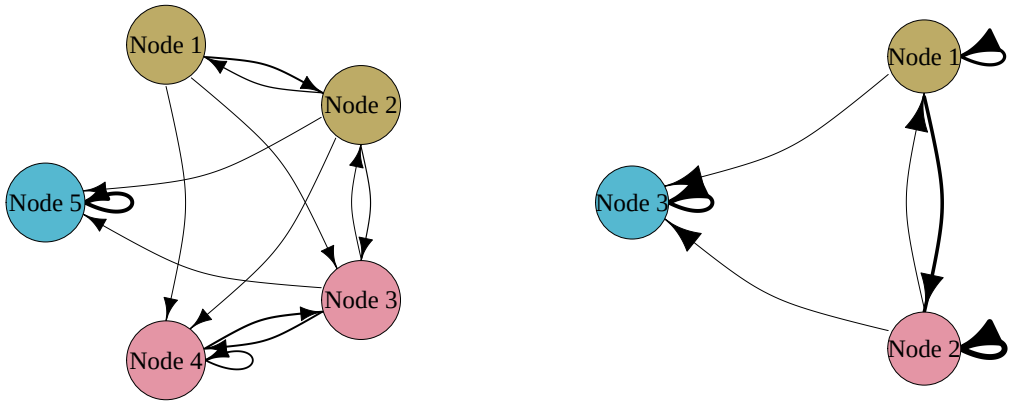

Figure 4-5: The graph on the right is formed from the graph on the left. Each cluster was merged into a single node, and the edges for that node were summed.

4.3 Application to the Gulf of Maine

4.3.1 Biophysical Model

Each of the clustering algorithms takes one or more connectivity matrices as its input, and we estimated the connectivity patterns using the biophysical model described in chapter 3. The model was forced using hourly archived output from the Finite Volume Community Ocean Model (FVCOM). FVCOM is a data-assimilative, free-surface, 3D model that solves the finite volume form of the primitive equations on a variable resolution mesh (Chen et al., 2006). The FVCOM output used in this study was generated using the 3^{rd}-generation mesh for the Gulf of Maine, which spans from Cape Hatteras to Nova Scotia and represents the vertical structure using 45 σ-layers. Additional details about the FVCOM model configuration and validation are provided in chapter 5. In addition to the FVCOM output, we used a sediment composition database from Poppe et al. (2005) to classify the bottom substrate as fine sand, coarse sand, gravel, or bedrock.

The biological components of the model were developed in support of the trait-based modeling study described later in this dissertation. We describe them in detail in chapter 5 and highlight the most relevant aspects of them here. Briefly, we simulated 4 species that span a broad range of dispersal strategies: sea scallops, haddock, yellowtail flounder, and Atlantic herring. Whereas sea scallops and herring spawn their young over gravel substrate, yellowtail flounder spawn over sand, and haddock spawn over a variety of substrates. Spawning occurs in early spring for haddock, late spring for scallops and flounder, and fall for herring. Flounder and haddock larvae drift passively in our model, but scallop larvae actively swim towards to pycnocline,

and herring larvae engage in diel vertical migration. The larvae drift for periods ranging from as short as 30 days for sea scallops to as long as 240 days for herring larvae before settling. Settlement occurs as a stochastic process. The settlement probability varies among species and is a function of water depth and bottom substrate type. The parameters for the 4 species are listed in Appendix F, the processes are detailed in chapter 5, and the process of parameterizing the model is presented in Appendix E.

The IBM was used to generate high resolution connectivity matrices for each of the 4 species in 1995. Each connectivity matrix gives the transport probabilities among 10 km x 10 km square grid cells in the Gulf of Maine and surrounding areas. Each matrix was estimated using the procedure from chapter 2 so that each transport probability that was at least 5% likely to be greater than 0.01 was estimated so that the coefficient of variation was no larger than 0.1. Equivalently, using the notation of chapter 2, we set $\pi = 0.05$, $\delta = 0.01$, and $\epsilon = 0.1$. Particles were tracked using a timestep of 10 minutes and the particle states were archived daily. At the end of each timestep, particles were settled based on the output of a random number generator and the settlement probabilities. Particles that settled were frozen in their current position for the remainder of the simulation and the settlement time was noted.

4.3.2 Clustering

We used Infomap, the modified Louvain method, and the multigraph method to identify coherent clusters from the connectivity data for each species. Prior studies that have attempted to cluster population connectivity data suggest sweeping across a range of values for the tuning parameter to explore a variety of possible cluster counts and sizes (Jacobi et al., 2012; Thomas et al., 2014). In theory, this approach should work well because the detected clusters will become larger and fewer as the Markov time increases and as γ decreases. In practice however, we find that this relationship does not necessarily hold for population connectivity data. We propose an alternative method for parameterizing Infomap and the modified Louvain method, and we use this method to compare and contrast the results between the clustering algorithms.

Thomas et al. (2014) suggest that studies using the modified Louvain method sweep across a range of values for γ, and Figure 4-6 presents the results of this sweep for our study system. At small values of γ, there are few clusters, and these clusters contain nearly all of the nodes. As γ increases, the number of clusters increases, and the number of nodes included in these clusters decreases. At large values of γ, the number of nodes included in clusters continues to fall, but the number of clusters declines as well. This result is expected based on the properties of the CPM. At low values of γ, many connections are larger than γ, so many nodes are aggregated into a few clusters. At high values of γ, most connections are less than γ, so there are few connections strong enough to drive aggregation into clusters. These results are not sensitive to the threshold chosen to identify non-trivial clusters.

In contrast, Infomap does not exhibit clear relationships between the number of clusters detected and the Markov time or between the size of the detected clusters and the Markov time (Figure 4-7). At long Markov times, Infomap generally identifies

only a single cluster. For three of the species, the cluster contains approximately half of the spawning nodes, and for Atlantic herring, it contains only a few nodes. As the Markov time decreases, the number of identified clusters and number of nodes within these clusters generally increase, but the relationships are not nearly as smooth as for the modified Louvain method. The relationship may be moderately smoothed by considering only clusters containing at least 10 nodes as non-trivial. However, even the smoothed relationships are not predictable and monotonic, so setting the Markov time based on the number or size of the clusters is unlikely to be reliable for Infomap.

An alternative metric for measuring the strength of clustering is the proportion of edges that originate and terminate in the same cluster. Viewing each cluster as a subpopulation, this metric is equivalent to the commonly used self-recruitment rate in marine ecology. Previous studies that apply clustering methods to marine population connectivity data have used this metric by definition (Thomas et al., 2014) or referring to it as the coherence ratio (Rossi et al., 2014). For both the modified Louvain method and Infomap, the relationship between the log-transformed value of the tuning parameter and the coherence ratio is a sigmoid function (Figure 4-8).When the Markov time is sufficiently small, the pathway followed by each random walker in the Infomap algorithm effectively becomes random, and the coherence ratio jumps to a constant value of approximately 0.7. Based on these relationships, we suggest that researchers choose a desired coherence ratio based on the ecological goals of their study, then compute parameters for Infomap and the modified Louvain method based on this target value. Using this procedure, they may identify clusterings under each method that are equivalent according to an ecologically relevant metric.

Using this approach, we identified equivalent clusters using Infomap and the modified Louvain method. Overall, the clusters identified by both methods were similar (Figure 4-9). When γ and the Markov time were chosen so that the coherence ratio was near 0.25, the modified Louvain method identified 9 non-trivial clusters, and Infomap identified 6 clusters. The clusters were structured similarly, except that the modified Louvain method identified a cluster on Georges Bank and split the Eastern Maine Coastal Current (EMCC), Bay of Fundy, and southwest of Nova Scotia into separate clusters, but Infomap did not identify a cluster on Georges Bank and grouped the other regions into a single cluster. When we increased γ and reduced the Markov time so that the coherence ratio was approximately 0.50, the number of identified clusters decreased and the size of each cluster increased. In this case, the modified Louvain method separated Georges Bank into a unique cluster and grouped Massachusetts Bay together with the Western Maine Coastal Current (WMCC), but Infomap identified a unique cluster in Massachusetts Bay and grouped Georges Bank together with southern New England (SNE). In this case, Infomap again separated Massachusetts Bay into its own cluster with one cluster each to the north and south of it. The modified Louvain method instead identified only two clusters and placed the boundary between them at the approximate division between the EMCC and WMCC. In each of our test cases, the overall patterns identified by each method were similar, but differences between them did exist.

We also used the modified Louvain method to identify clusters for each of the four species individually (Figure 4-10). The individual species plots revealed that the

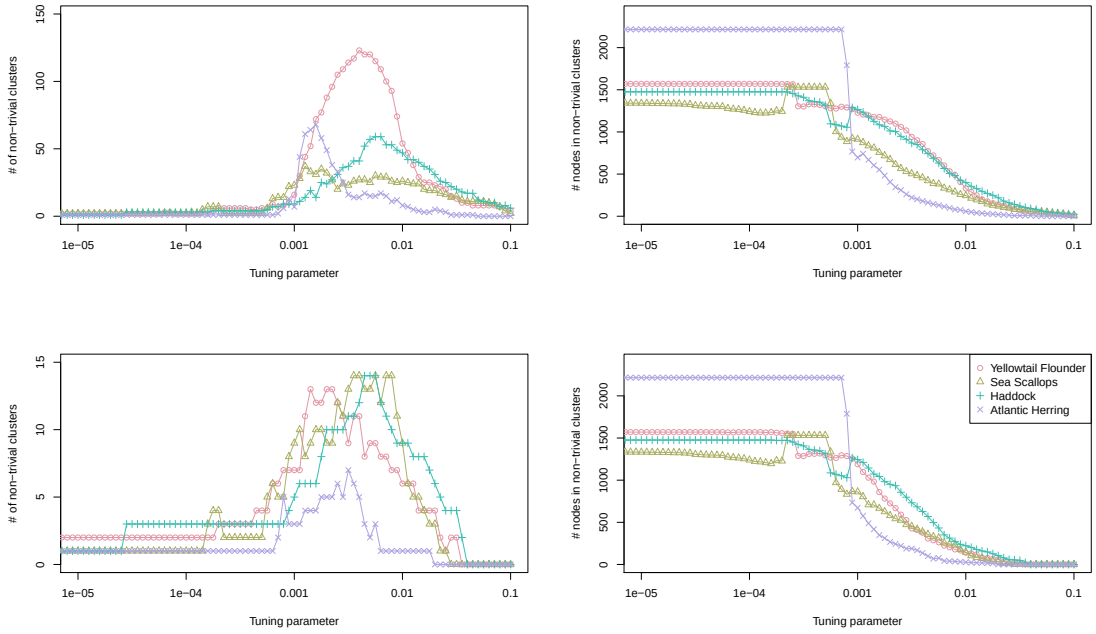

Figure 4-6: The number of non-trivial clusters (left column) and number of nodes belonging to these clusters (right column) is plotted as a function of the tuning parameter, γ, for the modified Louvain method. A non-trivial cluster was defined as a cluster containing at least 2 nodes in the top row and a cluster containing at least 10 nodes in the bottom row. The color and symbol used for plotting indicates the species for which the clustering algorithm was run.

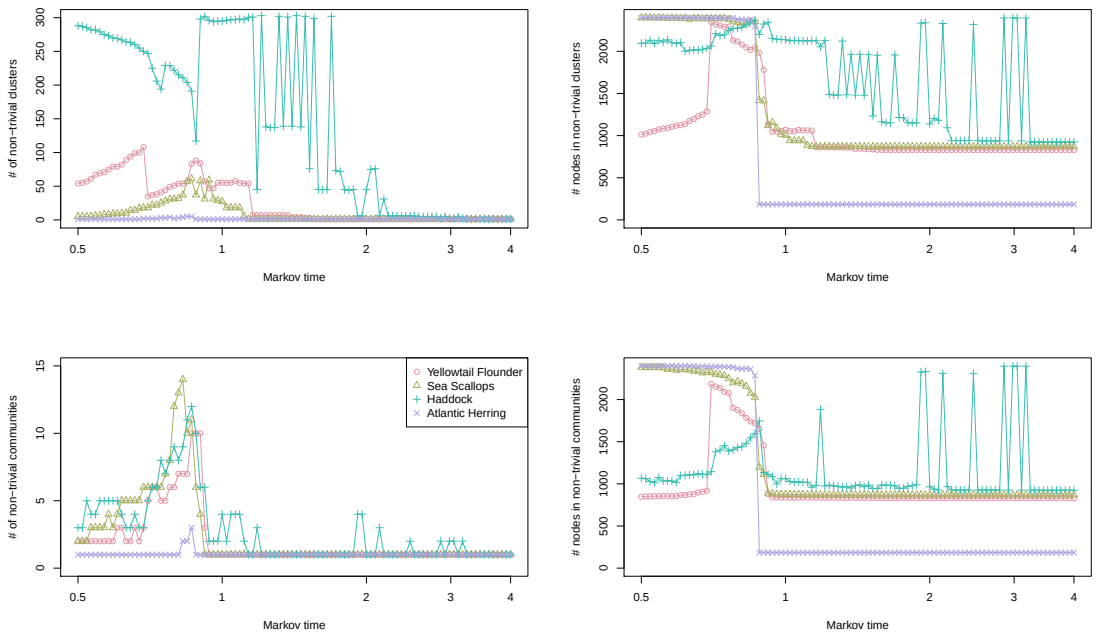

Figure 4-7: The number of non-trivial clusters (left column) and number of nodes belonging to these clusters (right column) is plotted as a function of the Markov time for the Infomap algorithm. A non-trivial cluster was defined as a cluster containing at least 2 nodes in the top row and a cluster containing at least 10 nodes in the bottom row. The color and symbol used for plotting indicates the species for which the clustering algorithm was run.

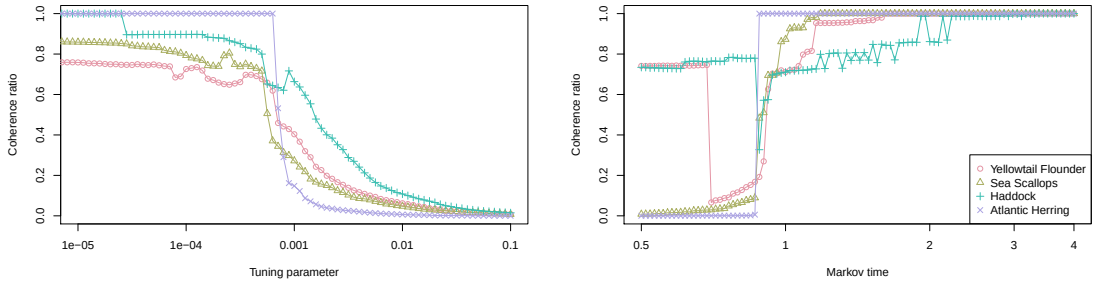

Figure 4-8: The average coherence ratio across all clusters is plotted as a function of γ for the modified Louvain method (left) and the Markov time for the Infomap algorithm (right). The color and plotting symbol indicates the species for which the algorithm was run.

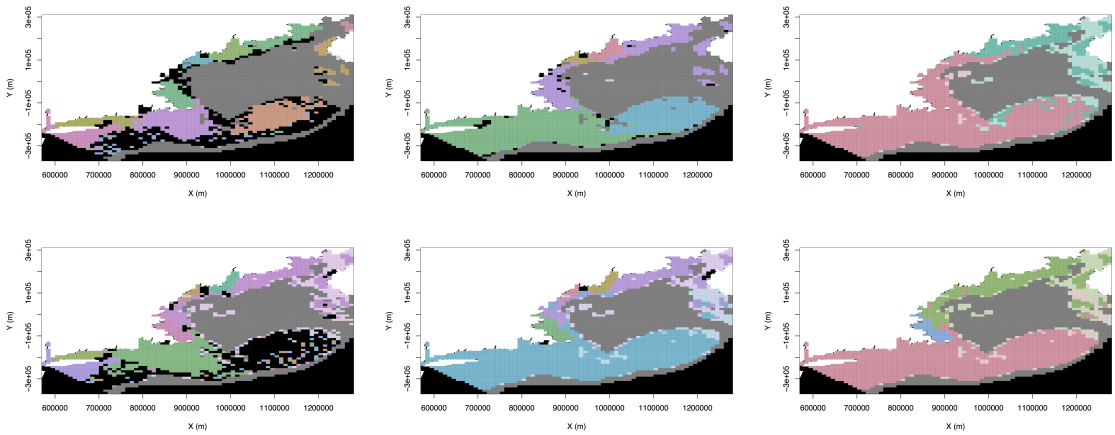

Figure 4-9: The clusters as identified by the modified Louvain method (top row) and Infomap algorithm (bottom row) for yellowtail flounder are plotted. The left column depicts clusters that were chosen to achieve a mean coherence ratio of 25%, the center column is for 50% and the right column for 75%. Areas on land are plotted in white, each color corresponds to a different cluster, and black areas are areas that were not clustered together with at least one other node including water that is outside of the study area. Spawning areas for the species are plotted in full color, and non-spawning areas are plotted in a lighter shade.

clustering patterns differ between the species. Although the habitat requirements and species traits for yellowtail flounder and haddock are similar, the clusters identified for haddock are generally smaller than those for yellowtail flounder. In addition, the modified Louvain method applied to haddock splits SNE into separate clusters for the inner and outer shelf in the 50% coherence ratio case and places the boundary between the two clusters at Cape Cod for the 75% coherence ratio case. For yellowtail flounder, it does not split SNE and places the split between clusters further north. The species that spawn exclusively over gravel have far less spawning than settlement habitat, and the clusters are much more patchy and scattered as a result. We revisit the spatial structuring of the clustering patterns in more detail in chapter 5.

Finally, we used the multigraph method to group the species into regimes. For each of the three coherence ratios, we applied the multigraph method using two different tuning parameters for the regime clustering step. The smaller value of the tuning parameter was chosen so that three species would be joined into a single regime with a single species excluded. This procedure resulted in tuning parameter values of 0.80, 0.70, and 0.60 for the 25%, 50%, and 75% coherence ratios. In the 25% case, Atlantic herring was clustered separately from the others. In the other two cases, haddock was clustered separately from the others. The regime averaged clustering patterns for the regimes with 3 species are similar to those for the individual species, but generally depict fewer clusters (Figure 4-11). The second tuning parameter was chosen so that two species would cluster into a single regime, and the values used were 0.85, 0.75, and 0.65 for the three coherence ratios. For the 25% and 50% coherence ratio cases,

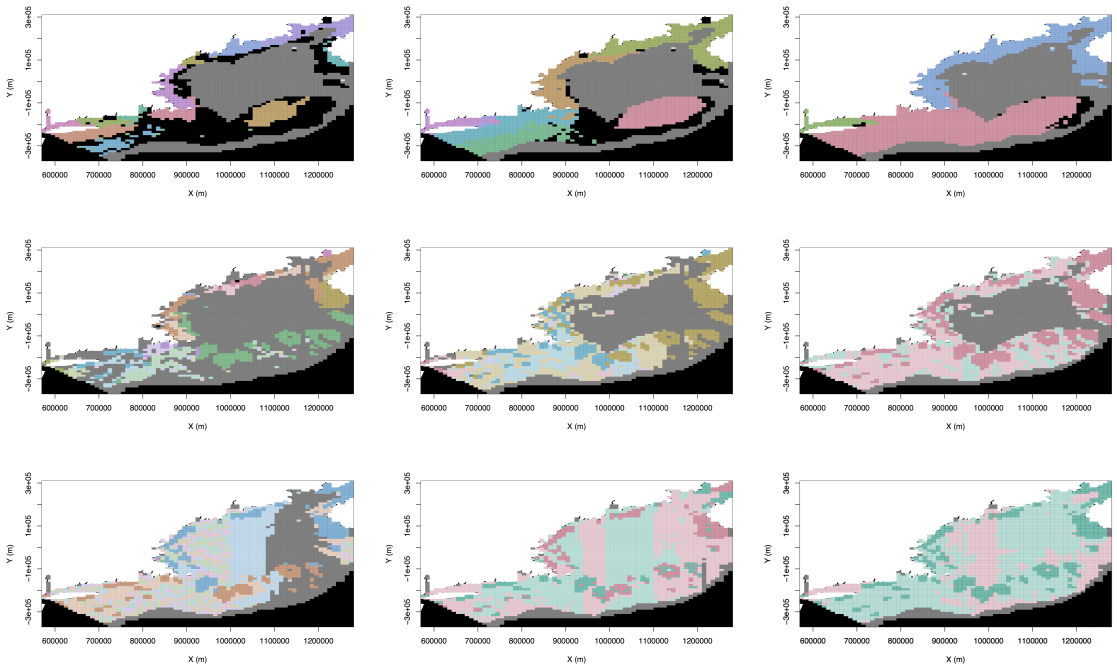

Figure 4-10: The clusters as identified by the modified Louvain method for haddock (top row), sea scallops (center row), and Atlantic herring (bottom row) are plotted. The left column depicts clusters that were chosen to achieve a mean coherence ratio of 25%, the center column is for 50% and the right column for 75%. Areas on land are plotted in white, each color corresponds to a different cluster, and black areas are areas that were not clustered together with at least one other node, including water that is outside the study area. Spawning areas for the species are plotted in full color, and non-spawning areas are plotted in a lighter shade.

the sand spawning species (yellowtail flounder and haddock) clustered together and the gravel spawning species (sea scallops and Atlantic herring) clustered together. In the 75% coherence ratio case, yellowtail flounder clustered together with Atlantic herring and the other species each formed a unique regime.

4.4 Discussion

Overall, both the modified Louvain method and Infomap identify similar structuring patterns, but the interpretation of the patterns differs between the methods. The Infomap algorithm uses the pathway followed by random walkers through the graph to quantify the connectivity patterns, which simulates multiple generations of movement with time-invariant connectivity patterns and each individual producing exactly one offspring that survives to reproduce. This process captures multiple step transitions and weights each node according to the amount of time that it is visited. In contrast, the CPM examines only a single level of connections and simulates the transport process during a single generation.

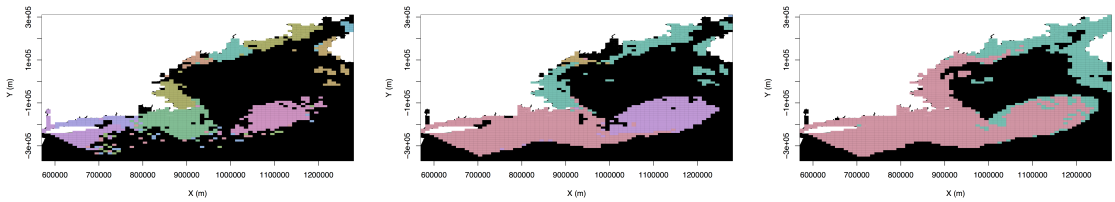

Figure 4-11: The clustering patterns identified by the multigraph algorithm are plotted here. In each case, three different species are included in the regime. The tuning parameter for the individual species clustering step was specified to obtain a coherence ratio of 25%, 50%, and 75% in the left, center, and right plots respectively. In each case, areas on land are plotted in white, each color indicates a different cluster, and areas depicted in black do not belong to a cluster.

Although the algorithms examine different processes, the clustering patterns that they identify are similar. This result reassures us that the clustering patterns that emerge are primarily driven by the underlying connectivity data, and not by the researcher's choice of algorithm. In most cases, most of the breaks between clusters aligned well with stock structures that have been identified by genetic and other studies. Van Wyngaarden et al. (2017) examined genetic differentiation among scallop populations in the northwest Atlantic. They found that within US waters, scallops are relatively well mixed, but that a boundary between stocks appears off the coast of Nova Scotia. Comparing 4 sites along the Maine coast and one on Georges Bank, Owen and Rawson (2013) found more genetic differentiation between the Georges Bank site and the other sites than among the sites on the Maine coast. Kenchington et al. (2006) also found that scallops on Georges Bank are distinct from those in other regions in the Gulf of Maine. Cadrin (2010) found a similar result from an interdisciplinary analysis of stock structure of yellowtail flounder. He concluded that three separate stocks exist in the northeastern US waters: one around Cape Cod and in the Gulf of Maine, a second on Georges Bank, and a third for southern New England and the Mid-Atlantic region. Our results followed a similar pattern, and one of the prominent boundaries between clusters that remained even at higher coherence ratios was between the Maine Coastal Current and southern New England.

Although the results from the different clustering algorithms were similar, there were some differences between them. Most notably, the modified Louvain method was more likely to identify a cluster on Georges Bank, and Infomap was more likely to identify one in Massachusetts Bay. One potential hypothesis for why this difference emerges is that Massachusetts Bay is likely a transitory location that receives particles from the Maine Coastal Current and disperses particles to SNE. Because Infomap simulates random walkers that visit multiple nodes, keeping the node names short for transition nodes is important to minimizing the objective function. If Massachusetts Bay is indeed a transition cluster that links other regions, then one way to optimize the the objective function would be create a small cluster with all short node names. Georges Bank is along the edge of the domain and less likely to be a transition

cluster between nodes. The modified Louvain method only considers single step connections, and so it is more likely to cluster Massachusetts Bay with the region to which it is more strongly connected (i.e. include it in either the Maine Coastal Current or SNE cluster). This hypothesis could be tested by computing the graph theoretic metrics betweenness centrality and eigenvalue centrality for each cluster. Clusters that are frequently visited by Infomap walkers would have high values of either metric. Unfortunately, this hypothesis was not supported by computing the centrality metrics, so additional research into this topic may be warranted.

Our model simulated dispersal using a simplified representation of transport during the larval phase, which excludes other processes that may be important in determining transport patterns. One of the differences between our study and prior ones that apply Infomap or the modified Louvain method to population connectivity data is that our species may have different spawning and settlement sites. When computing the connectivity matrices, we estimated movement from settlement to spawning sites by assuming that particles are equally likely to move to any spawning site. We tested the sensitivity of the clustering algorithms to this assumption and found that it did not substantially change the results for most of the species. However, for Atlantic herring, this assumption resulted in odd clustering patterns. In particular, for the threshold chosen to result in a 50% coherence ratio, the boundaries between clusters were straight vertical lines in many cases (Figure 4-10). These odd boundaries most likely occurred because many of the 10 km x 10 km grid cells are settlement sites only for this species, so all of the transport probabilities originating from these cells were filled with an identical value. Each of these cells therefore contain highly similar transport probabilities and are treated nearly identically by the clustering algorithm. As a result, the boundary between clusters was predominantly driven by the order in which the cells were processed by the clustering algorithm for Atlantic herring, and the cells that were processed around the same time would be more likely to be placed in the same cluster than those processed later. Our algorithm processes the cells in columns moving from south to north before moving onto the next column, so the boundaries between clusters are aligned between columns. Because 32%, 84%, and 92% of the cells where settlement took place were also spawning sites for scallops, flounder, and haddock respectively, versus 18% for herring, this issue did not impact the results for flounder and haddock and had less severe impacts on scallops than herring. One promising approach to alleviate this issue would be to estimate juvenile and adult movement patterns from field data, then use these estimates to compute a connectivity matrix for the full life cycle. In addition, we excluded or simplified processes such as growth, predation, and directed swimming, which may alter the transport patterns during the larval stage (Cowen et al., 2000; Paris et al., 2007; Petrik et al., 2014).

The multigraph method appears to do a good job of identifying which clustering patterns are most similar to one another and of maintaining the overall spatial patterns for each regime while averaging clustering patterns among the member species. Although the multigraph method was originally designed to consider the connectivity patterns from many species or from the same species during different years, our analysis here only considered 4 species during a single year. This restricted com-

parison allowed us to visually compare the clustering patterns from the individual species analyses against those from the multigraph method and assess the ability of the multigraph method. We look forward to exploiting the full potential of the multigraph method in the future using a sufficiently large number of species that visual comparisons alone would not be practical. Chapter 5 presents one such use.

Although we only examined the first level of clustering here, all three clustering algorithms result in a nested hierarchy of clusters. As with the Louvain method on which they are based, both the modified Louvain method and multigraph method result in few nested levels (Blondel et al., 2008). In our case, all three algorithms generally produced at most three levels of clusters.

Overall, we found that the three clustering algorithms give similar but not identical results. For single species studies, we suggest the use of both the modified Louvain method and Infomap together. Although the resulting patterns will likely be similar from both methods, the computational cost of the clustering algorithms is trivial compared to the cost of running a biophysical model, and differences in the results between algorithms may highlight areas that warrant further investigation. For multi-species studies, our multigraph algorithm is an effective way to compare clustering patterns among species. Although we developed it by extending the modified Louvain method, it could easily be modified to use other clustering algorithms (e.g. Infomap) as well.

Chapter 5

Trait-based modeling

Abstract

Identifying the drivers of population connectivity is a key component of understanding ecosystem function and predicting how it may respond to environmental perturbations. For many marine species, population connectivity is primarily influenced by the dispersal that occurs during a pelagic larval phase. The Gulf of Maine ecosystem contains a variety of species that exhibit a diverse set of larval dispersal strategies, and this study seeks to simulate many of these strategies. We develop a generalized representation of larval dispersal, and parameterize it to represent 4 real-world and 100 artificially generated species. Based on the output from the model, we find that the species traits responsible for determining the spawning and settlement habitat requirements are most influential for regulating larval dispersal success and patterns. We conclude with the recommendation that future work examine how these traits may be influenced by environmental variability.

5.1 Introduction

The marine ecosystem includes a complex and dynamic set of inter-related processes that vary in space, time, and between species. Spatial variability emerges both from geographic variability in fundamental processes such as growth, mortality, and reproduction and from the movement of individuals among geographically separated subpopulations (Cowen and Sponaugle, 2009). This movement is known as population connectivity, and for many marine species, it is largely controlled by a pelagic larval stage (Cowen and Sponaugle, 2009; Pineda et al., 2007). Temporal variability in population connectivity patterns results from fluctuations in environmental conditions, food availability, and circulation patterns among other processes (Cowen and Sponaugle, 2009). Because each species has a unique combination of spawning patterns, larval traits, and settlement requirements, the impact of these fluctuations may vary among species. Trait-based modeling (TBM) is a framework that seeks to understand how the individual traits that define each species interact with the biotic and abiotic environment to regulate ecological processes, including population connectivity (Barton et al., 2013). This chapter describes the application of trait-based

modeling to better understand how environmental conditions and life history traits interact to shape larval dispersal patterns in the Gulf of Maine and surrounding areas.

Larval dispersal patterns emerge from the complex interactions between individual larvae and the physical environment that they inhabit. Although the dispersal patterns are strongly driven by ocean circulation patterns, the traits that define each species also exhibit substantial influence over them. At the beginning of the dispersal stage, the distribution of spawning intensity in time and space determines the environmental conditions and circulation patterns to which their larvae are exposed, and the overall spawning strategy varies widely between species. Some species, such as the sea scallop *Placopecten magellanicus*, spawn within a short window of time following an environmental cue, presumably to expose all of their larvae to a favorable set of environmental conditions (Posgay and Norman, 1958; Hart and Chute, 2004). Other species such as haddock, *Melanogrammus aeglefinus*, spawn over a period of months (Cargnelli et al., 1999a). Haddock spawning takes place in the spring roughly in accordance with the spring phytoplankton bloom, but the months long window hedges against slightly mistiming spawning. However, it also exposes larvae to a range of food conditions and circulation patterns, some of which may be more or less favorable to successful survival and dispersal. Continuous spawning throughout the year is largely restricted to tropical species in locations without strong seasonal differences in environmental conditions.

Spatially, spawning may be restricted by adult habitat needs, the location of spawning aggregations, or requirements for egg survival. Sedentary species such as scallops do not travel long distances to spawn, and so spawning is restricted to habitat that is suitable for adult survival and reproduction. In contrast, Atlantic herring, *Clupea harangus*, are highly migratory fish that spawn demersal eggs over gravel substrates because the bottom roughness helps to retain the eggs until they hatch (Reid et al., 1999). Within our study region, many of the most important spawning areas lie on Georges Bank or in the Great South Channel, and a number of studies have examined the processes that lead to successful dispersal for larval spawned within this region. Tian et al. (2009a) simulated the dispersal of fall spawned sea scallop larvae from Georges Bank over an 11 year period from 1995-2005. They found that many of the larvae spawned between the 60 m and 100 m isobaths were retained over Georges Bank, and that 1995 was a particularly strong year for retention in the area (Tian et al., 2009a). Ultimately, they concluded that scallop dispersal is primarily determined by the circulation patterns and locations of spawning populations, and that the strong tidal mixing recirculation was a major reason for the strong retention during 1995. Gilbert et al. (2010) also simulated the dispersal of sea scallop larvae from Georges Bank, but concluded that spring spawned larvae were less likely to be retained than fall spawned ones and that pycnocline-seeking behavior increased the likelihood of retention. Boucher et al. (2013) investigated retention of haddock on Georges Bank during the period from 1995 through 2009. They found that retention during 1995 was above average, but not the highest observed during their study period.

Once larvae enter the water column, their interactions with the physical environment, predators, and prey determine their dispersal patterns. At least initially, marine

larvae are planktonic organisms that drift passively with ocean currents (Cowen and Sponaugle, 2009). Pelagic larval duration (PLD), or the amount of time that a larva spends in the water column before recruiting to a settlement site, has been identified as one of the most important determinants of dispersal patterns. Because marine larvae are primarily transported by ocean currents and the PLD is the time that larvae are exposed to these currents, PLD regulates the maximum distance that larvae may be dispersed (Shanks et al., 2003; Shanks, 2009). As a result, species with longer PLDs tend to be better connected over greater spatial scales and coherent clusters of habitat sites are larger for these species (Thomas et al., 2014; Ser-Giacomi et al., 2015).

However, an increasing amount of evidence shows that many marine larvae may substantially influence their dispersal patterns by swimming in either the horizontal or vertical directions (Paris et al., 2007; Staaterman et al., 2012; Staaterman and Paris, 2014). Diel vertical migration (DVM) is a commonly observed pattern where planktonic organisms rise to the surface at night, then sink to deeper depths during the day. DVM may be a mechanism to find prey at night and avoid predation during the day when light allows predators to see better at the surface, or it may be a retention mechanism that helps larvae remain in the vicinity of suitable settlement habitat (Stephenson and Power, 1988). Some larvae may also be able to sense the location of suitable settlement habitat and swim horizontally towards it later in their development (Dixson et al., 2008, 2011). Prior studies examining vertical movements within our study region have found that the impacts of swimming vary geographically and between species. Boucher et al. (2013) found that haddock larvae advected in 2-dimensions were least likely to be retained on Georges Bank when released at the surface of the water and that 3D transport increased retention. Churchill et al. (2011) found that dispersal success for cod larvae released in the western Gulf of Maine and tracked at 2.5 m water depth was strongly tied to the presence of downwelling conditions. Introducing DVM and pycnocline-seeking behavior has generally increased the proportion of larvae that successfully settle in prior studies. Churchill et al. (2011) found that DVM behavior increased transport success rates over a multi-year timespan, and Gilbert et al. (2010) found that pycnocline-seeking behavior increased larval retention. However, Gilbert et al. (2010) also found that the impacts of swimming varied geographically. In addition to the behaviors that we have listed here, there are a variety of others. The intensity and associated stimuli for each behavior may also change as each larva develops, and modeling swimming behavior may include a large number of highly species specific parameters.

As larvae transition out of the water column, habitat requirements restrict where larvae may settle. Many tropical larvae live their adult lives on coral reefs that punctuate an otherwise unsuitable oceanic environment. In this case, the settlement requirements are simple; larvae simply must return within a certain distance from a coral reef. However, the definition of suitable settlement habitat for the continental shelf of temperate regions is less clear. Habitat suitability in these regions is determined by a variety of factors, including water temperature, bathymetry, and substrate type. Recently, multiple studies have observed northward shifts in species distributions and tied these shifts to ocean warming and thermal tolerances (Perry

et al., 2005; Nye et al., 2009).

Finally, when using IBMs to simulate larval dispersal, the configuration of the model itself may influence the results. Previous studies have both compared different configurations of the same physical models to explore the impact of the inclusion or exclusion of physical processes on larval dispersal and compared different hydrodynamic models for a single region. Tian et al. (2009c) simulated scallop larval dispersal from Georges Bank using three different configurations of the same hydrodynamic model and found that models based on the residual flow or relying on a weak non-linearity assumption did not adequately represent the complex circulation patterns in the region. They suggest that studies in this region use fully non-linear hydrodynamic models driven by spatially realistic and time-varying forcing fields (Tian et al., 2009c). Hufnagl et al. (2017) recently compared IBM simulations of larval dispersal using 11 different hydrodynamic models of the North Sea and found high variability between the models. Ultimately, they concluded that the interannual trends in population connectivity patterns were similar between models, but the absolute values for each year differed widely. Therefore, in order to accurately and systematically explore how the traits that define each species influence dispersal patterns, a framework that simulates many species using the same modeling environment is necessary. Trait-based modeling (TBM) provides this framework.

Trait-based modeling involves defining a set of virtual species by specifying combinations of life history traits for each, then using a simulation to evaluate how the traits and the interactions between them determine ecological processes and ecosystem structure (Barton et al., 2013). In contrast to targeted studies that faithfully represent the details of a few study species, TBM studies often use a more restricted set of traits to represent a variety of species (Barton et al., 2013). The parameter values for each trait may be chosen from a limited range of plausible values for species in the region. Although the simplified representation used by TBM studies may provide a less accurate portrayal of connectivity patterns for any given species, it facilitates quantitative analysis of the role of individual traits in determining connectivity patterns and allows a thorough exploration of the trait space. The computational species in this exploration may be representative of real-world species or may be completely abnormal for the region. As a result, TBM studies may highlight both successful and unsuccessful dispersal strategies. The TBM framework has been successfully used for simulating phytoplankton (Follows et al., 2007; Ward et al., 2014), copepod community structure (Maps et al., 2012), and larval fish transport Treml et al. (2015).

Using a trait-based model, this study seeks to answer the following two questions. First, which traits and combinations of traits result in a high probability of successful dispersal. Specifically, we seek to identify traits that are generally associated with high success rates, regardless of the other trait parameters exhibited by the species. Species that exhibit these traits are less likely to be strongly impacted by environmental changes. Conversely, species with traits that are successful only in association with certain environmental conditions or other traits are more likely to experience changes to their dispersal patterns or success rates with environmental changes. We expect that species with a short PLD or with spawning that is broadly distributed in time and space will exhibit success rates and dispersal patterns that are only minimally

impacted by environmental conditions and other traits. In contrast, we expect that species with restricted spawning distributions and long PLDs will require specific larval behaviors to be retained over suitable settlement habitat and will be more sensitive to the spawning time and location.

Second, we seek to identify how the traits exhibited by each species influence subpopulation structure. This analysis includes two parts. First, we seek to identify coherent geographic regions from the population connectivity data. Second, we seek to identify which of these regions are most important for maintaining connectivity throughout the region. We expect that long PLD, broadly distributed spawning in time and space, and minimal or no larval behavior will result in fewer, larger regions. These populations will likely contain a greater diversity of habitat types, including both source and sink regions and more varied bathymetric and circulation features. The Gulf of Maine is a diverse ecosystem with a variety of successful dispersal strategies, and we explore these questions there.

5.2 Methods

We simulated the dispersal of marine larvae in the Gulf of Maine and surrounding areas using a coupled biological-physical model. Our model uses archived output from the Finite Volume Community Ocean Model (FVCOM, Chen et al., 2006) to represent the physical environment. Within this environment, we used the individual-based model IBM from chapter 3 to simulate larval transport in the Gulf of Maine and surrounding areas. We then summarized the dispersal patterns using the multigraph clustering algorithm from chapter 4. This study focused on the year 1995, which was chosen to be representative of a typical year for the Gulf of Maine based on the analysis in Li et al. (2014).

5.2.1 Physical environment

FVCOM is a data assimilative, free-surface model that solves the finite volume form of the primitive equations on a variable resolution triangular mesh. The output that we used relies on the 3^{rd}-generation mesh for the Gulf of Maine, which spans the region from Cape Hatteras to Nova Scotia with 48451 vertices and 90415 triangular elements (Figure 5-1). Velocity vectors are produced at the center of each element and are archived hourly, and the elements smoothly transition in horizontal resolution from 200 m near the coast and over steep bathymetry to 15 km in the central Gulf of Maine. Vertical structure was represented using a σ-coordinate system with 45 layers. FVCOM is forced with the tidal elevation at the open boundary and with surface momentum, heat, and moisture fluxes at the surface using output from the MM5 atmospheric model (Sun et al., 2013). Time stepping is accomplished using a modified 4^{th}-order explicit Runge-Kutta scheme (Cowles, 2008). This configuration has been extensively validated against observed data for the region (Chen et al., 2005, 2006; Cowles et al., 2008; Sun et al., 2013, 2016) and has been used to simulate ecological processes for a variety of marine species (Huret et al., 2007; Tian et al.,

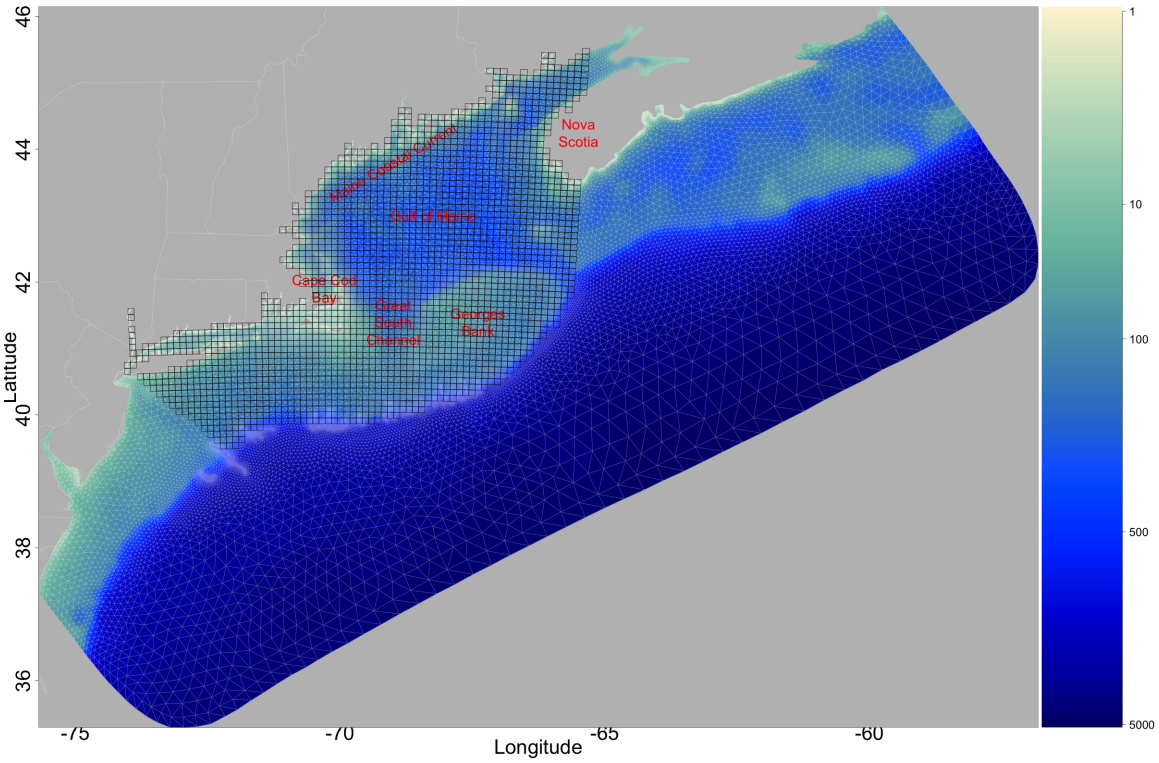

Figure 5-1: The Gulf of Maine and surrounding areas as represented by our model are plotted here. The background color indicates the bathymetry in meters. The white mesh overlaid on the water is the mesh used by FVCOM, and the black mesh is the 10 km x 10 km mesh used for calculating the connectivity matrix. Some important areas are noted in red text.

2009a; Boucher et al., 2013). In addition to the output from FVCOM, a sediment composition database from Poppe et al. (2005) was used to classify each FVCOM mesh element as fine sand, coarse sand, gravel, or bedrock (Figure 5-2).

5.2.2 Particle-tracking model

The proximate goal of the larval dispersal simulation was to estimate a connectivity matrix for each of the species under consideration. The connectivity matrices were computed using a regular 10 km grid to define the origins and destinations (Figure 5-1), and the sequential Bayesian procedure from chapter 2 was used to ensure convergence. We sought to estimate each transport probability that was at least 5% likely to be greater than 0.01 such that the coefficient of variation was no larger than 0.1. Equivalently in the notation of chapter 2, we set $\pi = 0.05$, $\delta = 0.01$, and $\epsilon = 0.1$.

All of the larval simulations were completed using the individual-based model (IBM) described in chapter 3. To briefly summarize, this model linearly interpolates environmental properties in both time and space, then uses a modified 4^{th} order Runge-Kutta algorithm and fixed timestep to compute the particle trajectories. For this study, we used a 10 minute timestep after testing that shorter timesteps did not

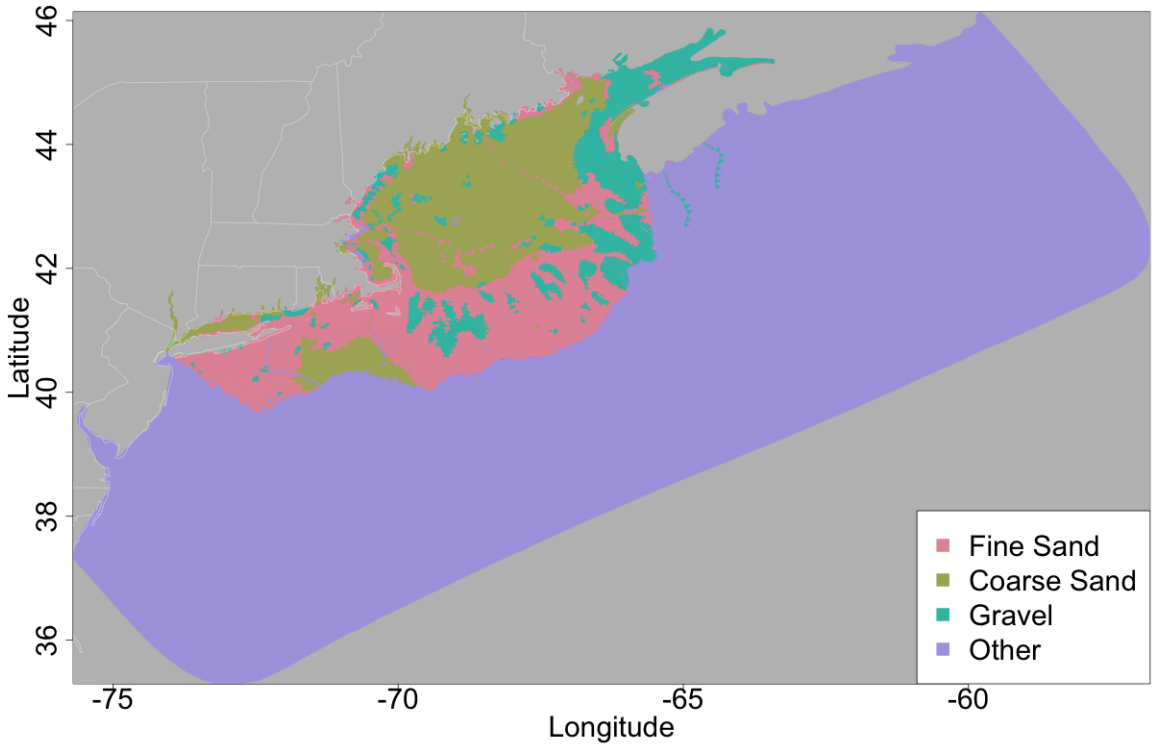

Figure 5-2: The Gulf of Maine and surrounding areas as represented by our model are plotted here. The color of each FVCOM element represents the substrate type for that element.

appreciably change the resulting trajectories. The spawning time and location for each particle were randomly drawn from the specified distributions for the species. The specification for the distribution of spawning time is discussed in subsection 5.2.3, and the distribution of spawning locations was uniform across all of the suitable habitat for the species within each 10 km by 10 km spawning cell. The trajectory of each particle was then integrated without including horizontal or vertical diffusion, but potentially including one of the vertical swimming behaviors described in subsection 5.2.3. Each species was assigned a unique, but fixed, pelagic larval duration (PLD) and competency window (CW), and larvae were eligible to settle beginning at PLD days age and ending CW days later. Settlement was modeled a probabilitistic process. At each timestep, larvae over suitable habitat settled with a fixed probability, and this probability was chosen so that a larva over suitable habitat for its entire CW had a 99% probability of settlement. The 99% probability was chosen to be large enough so that most of the particles eligible for settlement would settle, but small enough so that particles would not always settle immediately upon encountering suitable habitat. Allowing particles to transition over suitable habitat without settling was a necessary component of implementing settlement habitat preferences whereby a larva may be more likely to settle on some substrates than others. When each particle settled, it no longer moved for the remainder of the simulation and the time of settlement was noted.

5.2.3 Biological model

Our biological model is a generalized representation of marine larvae that can be parameterized to represent many species and includes the traits that we believe are most likely to influence larval dispersal in the Gulf of Maine. For this analysis, we simulated 4 real-world species and 100 artificially generated species that represent a diversity of life history strategies. The four real-world species are sea scallops, Atlantic herring, yellowtail flounder (*Pleuronectes ferruginea*), and haddock. The sea scallop is a benthic, bivalve species with a moderate length PLD and active vertical swimming behavior (Tian et al., 2009a; Gilbert et al., 2010). Although both spring and fall spawning has been recorded in our study area, we only simulate spring spawning (Hart and Chute, 2004; Gilbert et al., 2010). Atlantic herring larvae also swim vertically, but spawning occurs throughout the fall, the PLD is much longer, and juveniles recruit directly into the pelagic environment (Reid et al., 1999; Stephenson and Power, 1988). Whereas both scallops and herring spawn preferentially over gravel substrates and have behaviorally active larvae, the other two study species spawn over sand or a variety of habitat types and their larvae drift more passively (Reid et al., 1999; Cargnelli et al., 1999a; Johnson et al., 1999; Hart and Chute, 2004). Yellowtail flounder are more likely to be found over sandy bottoms, and haddock live over a variety of habitat types (Johnson et al., 1999; Cargnelli et al., 1999a). Overall, these four species exhibit 4 different reproductive strategies that are all successful in the Gulf of Maine.

The traits that we include in our model may be broadly split into spawning, larval, and settlement traits. We modeled spawning intensity as normally distributed in time

and restricted spawning locations based upon water depth and bottom substrate type. Haddock were modeled to spawn on 15 Mar±21 days (mean ± std. dev.) scallops on 15 May±7 days, flounder on 15 May±21 days, and herring on 15 Oct±21 days. All four species spawn over shallow banks and shelves, but the preferred bottom substrate varies by species. Both flounder and scallops were restricted to spawn in water less than 100 m depth, but scallops spawned only over gravel and flounder only over sand. These preferred substrates were chosen based upon the substrates where adults live. Haddock were likewise depth restricted to 90 m, but were allowed to spawn over either gravel or sand. Herring are a drastically different species from the other three in that they are pelagic and highly mobile, but migrate to spawning grounds. Presumably because they spawn demersal eggs and increased bottom rugosity helps prevent the eggs from washing away, herring spawn over gravel substrates in less than 80 m deep water. After spawning, each larva is advected for a species specific PLD, and in the case of scallop and herring, swims vertically towards a variable target depth. Scallop larvae actively swim towards the pycnocline where their food aggregates, which in the case of this study, was approximated by the mixed layer depth. The diel vertical migrations undertaken by herring larvae were simulated using the movement formulation from Zakardjian et al. (1999) and approximate depths from Stephenson and Power (1988). Finally, settlement was implemented as a probabilistic process as described above, and the origin site, destination site, and location of each particle were recorded every 6 hours. The parameters for each species are listed in Appendix F, and the process of choosing these four species is presented in Appendix E.

In addition to the 4 real-world species discussed above, we generated 100 artificial species. The parameters for these species were generated by randomly sampling from the range of plausible values for species in the Gulf of Maine. The process of generating distributions of plausible distributions for each trait is presented in Appendix E and the parameters for each species are reported in Appendix F. Although the sheer number of possible trait combinations makes it impossible to comprehensively evaluate the trait space, these randomly generated species provide insight into how individual traits and combinations of traits influence larval dispersal success rates and subpopulation structure.

5.2.4 Analysis

We computed 104 connectivity matrices P_s and analyzed these matrices to assess the drivers of both dispersal success and the spatial patterns of dispersal in the Gulf of Maine. Each element $p_{s,ij}$ of P_s gives the likelihood that a larva of species s that was released from origin cell i will settle in destination cell j.

To provide a baseline against which individual species can be compared, we first computed general statistics that summarize the number of larvae simulated for each species, likelihood of success for each larva, and expected destination for each larva. Because the simulations were configured to ensure convergence based on the algorithm from chapter 2, the distribution of particle releases was not uniform geographically or across species. To avoid potential biases from this property, we first normalized the connectivity matrices based on the assumptions that spawning intensity is uniform

across all suitable habitat for each species and that each species should be given equal weight in the analysis.

To identify the role of each trait in determining dispersal success, we used a regression analysis. Our regression model attempted to predict the proportion of larvae that would successfully settle for each species as a function of the trait parameters for that species. Visual inspection of the results revealed that some of the traits did not directly or linearly correlate with dispersal success, so we synthesized additional predictors that may be more meaningful. The maximum spawning and settlement depths for each species were drawn from lognormal distributions, so we log transformed those variables before using them. Dispersal success also showed an annual pattern, so we cosine transformed the mean spawning time for each species. The spawning time was originally expressed as the number of days since 1 Jan 1995. Finally, we computed the standard deviation of the settlement probability on gravel, coarse sand, and fine sand for each species. Because these settlement probabilities sum to 1, large values of this variable indicate strong preferences for a specific substrate type. The regression model was fitted using an ad hoc step up approach. Beginning with a regression that contained no predictor variables, we added the single predictor variable that showed the strongest relationship with the residual variance based on visual inspection. We repeated this process until there no longer appeared to be any relationships between the residual variance and any unused predictor variables or combinations of predictor variables.

To examine geographic structure in the connectivity patterns, we applied the multigraph clustering algorithm from chapter 4. The multigraph algorithm attempts to identify coherent geographic clusters from multiple graphs, where each graph represents the population connectivity patterns for a given species. In addition, it clusters the graphs together based upon the similarity in clustering patterns between each pair of graphs. It operates using two tuning parameters: γ, which regulates the number and size of the geographic clusters for each species, and χ, which regulates the number and size of the regimes when the species are clustered together. For this study, we chose three values of γ for each species so that each choice of γ would result in a target coherence ratio. The coherence ratio is the sum of all $p_{s,ij}$ that originate and terminate in the same cluster divided by the sum of all $p_{s,ij}$ for that species. The values of γ were chosen so that the coherence ratio would be 0.25, 0.5, and 0.75. For each target coherence ratio, we swept across a range of values for χ to explore how the species clustered together. Finally, we used the `rpart` function in R (R Core Team, 2016) to fit a classification tree that predicted the regime identity for each species based on the traits exhibited by that species.

Finally, as a brief sensitivity analysis to consider one possible impact of climate on our results, we explored how the overall probability of dispersal success and the results of the linear regression analysis would change if we excluded the southernmost 10% and 25% of the spawning cells for each species. For this sensitivity analysis, we removed any particles that were spawned within the southernmost cells from the results, and assumed that any particles that settled within these cells would not have otherwise settled. This procedure is not strictly accurate because the particles which settled in the southernmost cells may have otherwise moved north again and settled

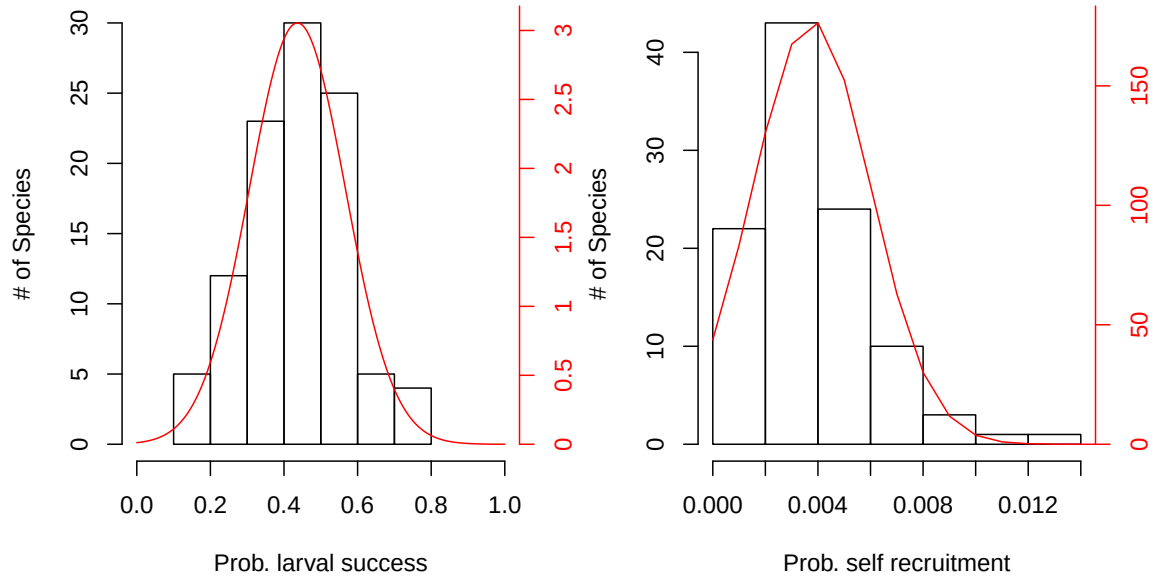

Figure 5-3: **Left:** This histogram depicts the probability that a larva will successfully settle for each species. The height of each bar indicates the number of species with a success rate contained within that bin. The red line is the probability density function for a normal distribution that was fitted to the data. **Right:** This histogram depicts the probability that a larva will successfully settle in the same 10x10 km grid cell where it was released for each species. The height of each bar indicates the number of species with a self-recruitment rate contained within that bin. The red line is the probability density function for a normal distribution that was fitted to the data.

elsewhere. However, the dominant flow direction at the southern range of our domain is towards the south, so this procedure gives a good initial approximation of the potential effect from northward species range contractions.

5.3 Results

Overall, we simulated a total of 959 million larvae. However, these larvae were not uniformly distributed across the 104 species, and were instead chosen to satisfy the termination criterion of the procedure from chapter 2. The number of larvae for each species ranges from a minimum of 1 million larvae to a maximum of 36 million larvae.

The distribution of dispersal success rates for each species may be approximated by a truncated normal distribution with mean 43.63% and standard deviation 13.06% (Figure 5-3). The self-recruitment rate for each species is the mean proportion of particles that settled in the 10 km by 10 km grid cell where they were spawned averaged across all grid cells. For our study, the self-recruitment rates were mostly

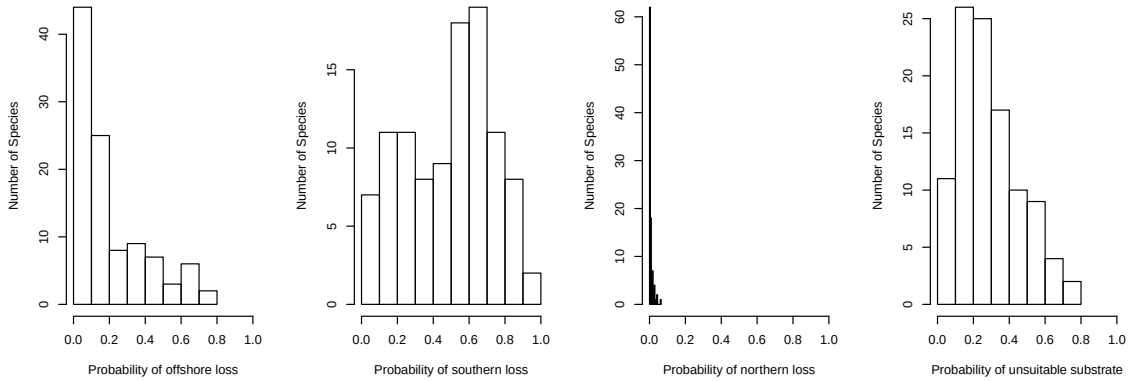

Figure 5-4: The probability of a particle being lost to the open ocean (far left), Mid-Atlantic Bight (center left), Scotian Shelf (center right), or unsuitable habitat within the study area (far right) is plotted here. The height of each bin indicates the number of species for which the probability of loss, given that a particle did not settle, falls into the indicated bin.

below 1% and ranged from less than 0.1% to 1.25%. The median dispersal distance for successful particles for each species was 125±46 km (mean±std. dev), so the low self-recruitment rates are unsurprising. Larvae that did not successfully settle may have moved offshore out of suitable habitat, south past the Hudson Canyon and out of the study area, north into the Scotian Shelf and out of the study area, or have remained within the study area but not found suitable habitat. For many of the species, unsuccessful larval settlement was most strongly driven by washout to the Mid-Atlantic Bight (Figure 5-4). A substantial number of larvae also remained within the Gulf of Maine area but were not over suitable settlement habitat, which suggests that retention of larvae near the spawning region alone is not sufficient to ensure successful dispersal. As could be expected from the regional circulation patterns, very few larvae moved north onto the Scotian Shelf.

5.3.1 Trait influences on dispersal success

In order to explain how the species traits determine the dispersal success rates, we fit a linear regression model. Overall, this model explained 62% of the inter-specific variability in larval success rates (Table 5.1) using 5 of the species traits together with 1 additional feature that we synthesized from these traits. Three of the species traits were used directly in the regression model, including the vertical swimming behavior, the spawning substrate, and the minimum PLD length. The cosine-transformed mean spawning date and log-transformed maximum settlement depth were also included in the model. The final predictor variable for our model was the the standard deviation of the settlement probabilities for fine sand, coarse sand, and gravel for each species. The two most influential predictor variables were the vertical swimming behavior and spawning habitat, and a regression using these two variables alone predicted 44%

Coefficient	Estimate	Std. Error	t-value	Pr(>|t|)	
Intercept	0.2980160	0.0851619	0.000715	3.499	***
Gravel only spawning	0.0799935	0.0203620	3.929	0.000163	***
Sand only spawning	-0.0034063	0.0205311	-0.166	0.868585	
Surface-tracking	-0.1728278	0.0228729	-7.556	2.70e-11	***
Pycnocline-seeking	0.0307545	0.0222325	1.383	0.169844	
DVM Behavior	-0.0104130	0.0228424	-0.456	0.649539	
cos(mean spawning time)	0.0362392	0.0115678	3.133	0.002309	**
$\log_{10}$(max settlement depth)	0.1779067	0.0411789	4.320	3.86e-05	***
Minimum PLD	-0.0021059	0.0005814	-3.622	0.000474	***
Std. dev in settlement probs	-0.3015379	0.0695159	-4.338	3.62e-05	***

Table 5.1: The fitted coefficients for a linear regression that attempts to predict the probability of larval settlement success are reported here. The regression was fit treating each species as an independent observation. * indicates significance at the 0.05 level, ** at the 0.01 level, and *** at the 0.001 level.

of the variability in the settlement probabilities. Relative to passively dispersing particles, surface-tracking particles were 17.3±2.28% less likely to successfully settle. Species that spawned over gravel only were predicted to have dispersal success rates 8.0±2.00% higher than species that spawned over either gravel or sand. The other spawning habitats and swimming behaviors had non-significant impacts in the model. Unsurprisingly, species with a greater maximum settlement depth, shorter PLD, or more uniform settlement habitat preferences were predicted to have higher success rates. The coefficients for all of the predictor variables are reported in Table 5.1. Although we examined the possibility of including interaction terms between variables, doing so did not improve the quality of the fit so they were not included in the final model.

5.3.2 Geographic structure

To explore how the species traits give rise to geographic structure, we applied the multigraph clustering algorithm using three different sets of values for the tuning parameter γ and sweeping across a range of values for χ. Regardless of how γ was specified, the number of regimes increased from 1 when γ was small to 104 when γ was large (Figure 5-5). The value of χ at which multiple regimes formed varied between the choices of coherence ratio, indicating that the clusters identified by the larger coherence ratios were more similar between species than those for the smaller coherence ratios. The result reflects that the broad scale dispersal patterns are more strongly driven by circulation patterns, but the fine scale details may be more strongly influenced by species traits. The distribution of species between regimes was highly uneven and most species were included in only a few large regimes. This feature was particularly apparent when there were only a few regimes, the largest regime contained most of the species.

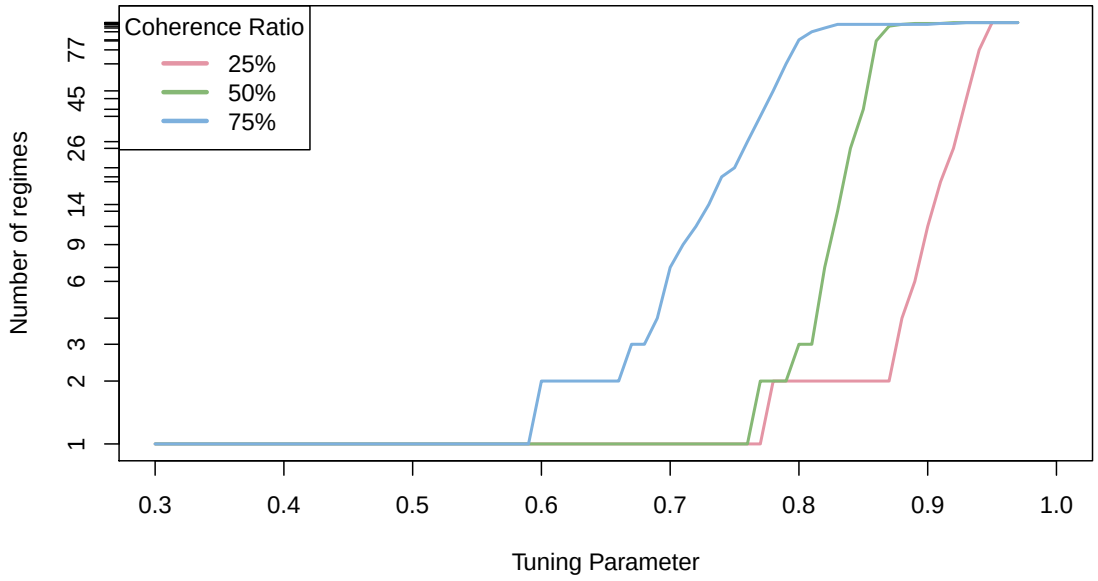

Figure 5-5: The number of regimes identified by the multigraph algorithm is plotted as a function of the tuning parameter, χ. The color of each line indicates the coherence ratio that was used to choose γ. The values of χ tested span from 0.30 to 0.97 with a uniform spacing of 0.01.

We chose two values of χ for the 75% coherence ratio case and examined the regimes in detail to better understand how species traits influence the clustering patterns at broad spatial scales. The first value chosen, $\chi = 0.69$, resulted in 4 regimes and was chosen to highlight the traits the resulted in vast differences between the clustering patterns. The largest regime contained 73 of the 104 species and resulted in a single cluster that contained nearly all of the spawning habitat (Figure 5-6). A second regime containing 25 species identified one large cluster containing Southern New England (SNE) and Georges Bank (GB), two smaller ones containing the New England coast and the Canadian coast, and a fourth along the shelf break of SNE. Two smaller regimes containing 2 and 4 species also emerged.

The second value chosen, $\chi = 0.73$, resulted in 14 regimes, many of which contained only one or a few species (Figure 5-7). This value was chosen to provide detail into the traits that result in specific boundaries between clusters. For instance, regimes 8 and 14 both contain a single cluster that encompasses nearly the entire spawning area for the species. However, whereas the cluster for regime 14 includes Cape Cod Bay and the waters south and east of Long Island, the cluster for regime 8 does not include the areas off of Long Island and Cape Cod Bay forms a separate cluster. Likewise, regimes 3, 5, 9, and 10 all depict clustering patterns with two large clusters. The break between these clusters for regimes 5 and 10 occurs at Cape Cod, but regime 10 includes substantially less habitat and is restricted to near the coast. For clusters 3 and 9, GB is included with the northern cluster and SNE forms a separate cluster. Regimes 6, 7, and 13 all depict patterns with 3 clusters, but the layout of these clusters differs widely between regimes. Regimes 6 and 13 include nodes in SNE, GB, and along the coast of New England and Canada, but regime 7 also includes a vast portion of the central Gulf of Maine. Regime 6 also separates the coast of Massachusetts, New Hampshire, and southern Maine into a distinct cluster, but regime 13 includes these areas within the SNE and GB cluster.

To better understand how the clustering patterns were driven by the species traits, we visually examined the distribution of trait values for each regime and attempted to predict the regime for each species using a recursive partitioning classification tree. This analysis focused on the regimes identified by the multigraph method using our second value for χ, 0.73. In total, the classification tree contained 11 binary splits (Figure 5-8). Each split considered only a single species trait and the splits were nested to a maximum depth of 6 splits. Five of the 11 splits considered the maximum spawning depth and 3 considered the maximum settlement depth. The spawning substrate, mean spawning time, and vertical swimming behavior were each considered for 1 split. Overall, the classification tree correctly predicted the regime for 83 of the 104 species (80.5%). Of the 50 species in regime 14, the largest regime, 43 were partitioned from the other species on the basis of a maximum settlement depth deeper than 84.43 m, a maximum spawning depth shallower than 175.4 m, and a mean spawning time earlier than Nov 26. Eight other species were classified in this group, including all three of the species in regime 1. The species in regime 1 were partitioned from those in regime 14 based on their surface-tracking behavior. Regime 8, which exhibited similar clustering patterns to regime 14 except in the Cape Cod Bay and SNE areas, was generally distinguished from regime 14 by having deeper

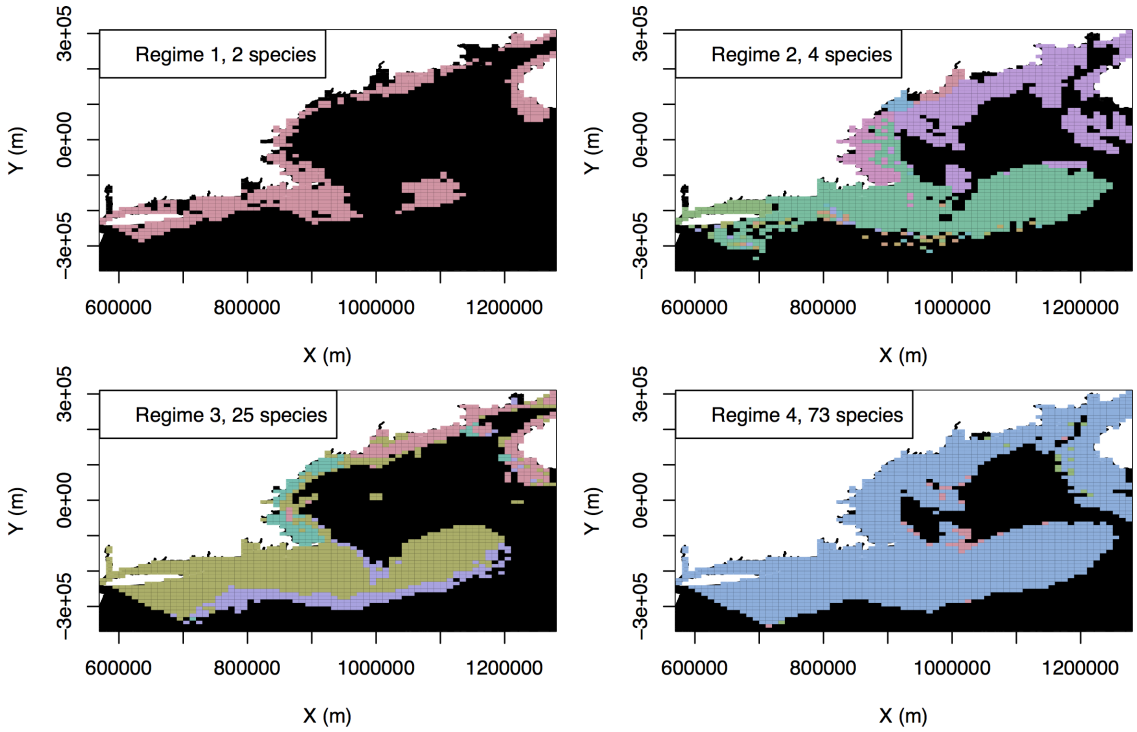

Figure 5-6: The clusters are plotted for each of the 4 regimes detected by the multi-graph method. The tuning parameter γ was set such that the coherence ratio for each species was 75% and χ was set to 0.69. Each color indicates a different cluster, white areas are land, and black areas were not part of a non-trivial cluster. Non-trivial clusters were defined as those containing at least 10 nodes, and the number of species within each regime is noted within each subplot.

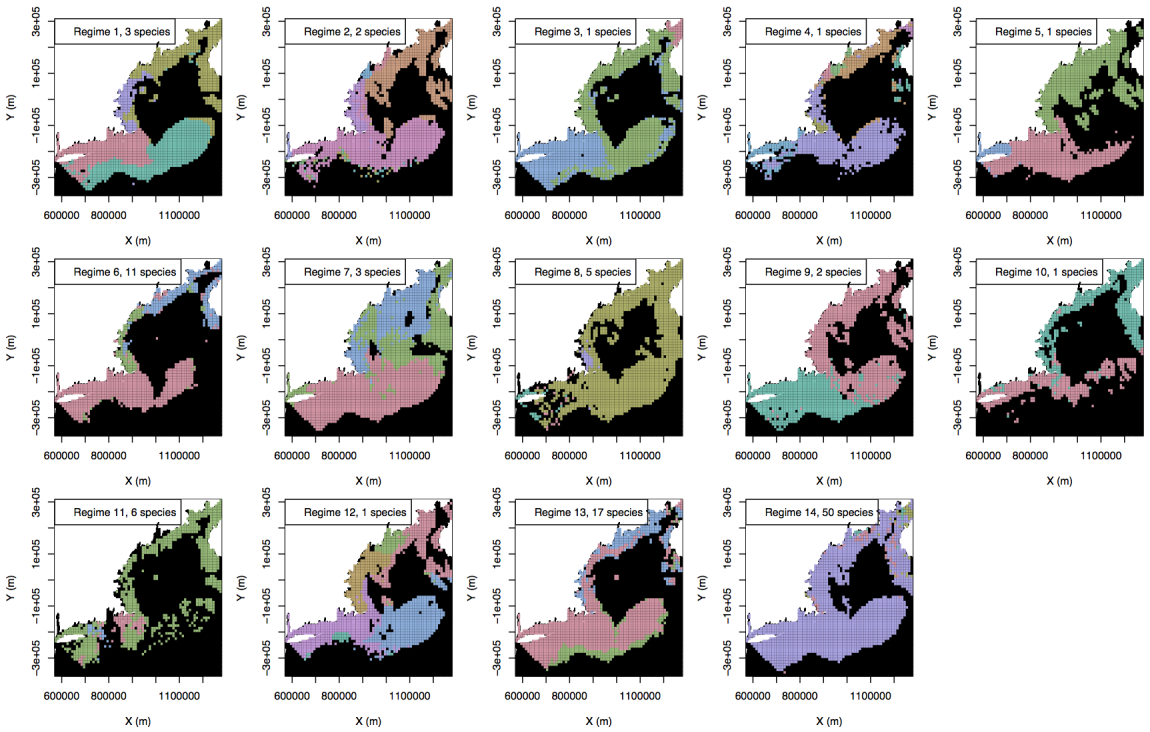

Figure 5-7: The clusters are plotted for each of the 14 regimes detected by the multigraph method. The tuning parameter γ was set such that the coherence ratio for each species was 75% and χ was set to 0.73. Each color indicates a different cluster, white areas are land, and black areas were not part of a non-trivial cluster. Non-trivial clusters were defined as those containing at least 10 nodes, and the number of species within each regime is noted within each subplot.

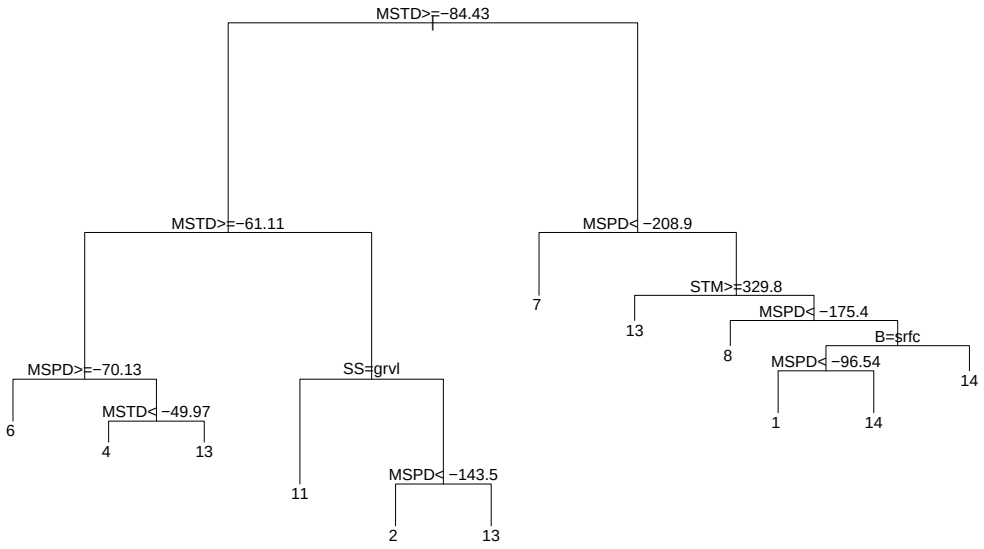

Figure 5-8: The classification tree for the regimes depicted in Figure 5-7 is plotted here. In each case, nodes that satisfy the condition listed at each split are listed under the left branch and nodes that do not satisfy the condition are under the right branch. The most likely regime for each terminal leaf is listed below the leaf. The variables and units for each are maximum settlement depth (MSTD, m), maximum spawning depth (MSPD, m), spawning substrate (SS, grvl=gravel-only), spawning time mean (STM, yearday), and behavior (B, srfc=surface-tracking).

spawning depths. Regime 6 and 13, which were generally similar but differed in their clustering of the coast of central New England, were partitioned largely based on their settlement depth. Ten of the eleven species in regime 6 had maximum settlement depths shallower than 61.11 m, but 14 of the 17 species in regime 13 were deeper than this depth. However, 11 of the species in regime 13 had maximum settlement depths between 61.11 m and 84.43 m, highlighting that even slight changes in this parameter may change the clustering patterns.

5.3.3 Sensitivity Analysis

Based on the results of our sensitivity analysis, the mean proportion of larvae that successfully dispersed, averaged across all of the species, decayed from 43.6% when all of the habitat was included to 11.3% when the southernmost 80% of the grid cells were excluded from the simulation. The decay was approximately linear, but there was initially little change when 0-15% of the cells were excluded, followed by a more rapid decay when 15-25% of the cells were excluded. This result is most likely due to a boundary effect where larvae from the southernmost grid cells would have

Coefficient	Estimate	Std. Error	t-value	Pr(>|t|)	
Intercept	0.2522567	0.0783118	3.221	0.00175	**
Gravel only spawning	0.0293697	0.0187241	1.569	0.12011	
Sand only spawning	-0.0465554	0.0188796	-2.466	0.01548	*
Surface-tracking	-0.0690678	0.0210331	-3.284	0.00144	**
Pycnocline-seeking	-0.0038768	0.0204442	-0.190	0.85001	
DVM Behavior	-0.0307991	0.0210050	-1.466	0.14591	
cos(mean spawning time)	0.0154313	0.0106374	1.451	0.15020	
$\log_{10}$(max settlement depth)	0.1640898	0.0378666	4.333	3.68e-05	***
Minimum PLD	-0.0024413	0.0005346	-4.566	1.50e-05	***
Std. dev in settlement probs	0.1640898	0.0378666	4.333	3.68e-05	***

Table 5.2: The fitted coefficients for a linear regression that attempts to predict the probability of larval settlement success are reported here after removing the southernmost 25% of the spawning cells for each species. The regression was fit treating each species as an independent observation. * indicates significance at the 0.05 level, ** at the 0.01 level, and *** at the 0.001 level.

been more likely to disperse to areas south of the study area. After removing the southernmost 10% of the grid cells for each species, the coefficients for the linear regression presented in Table 5.1 were nearly unchanged, and there was substantial overlap in the 95% confidence interval for all of the coefficients. After removing the southernmost 25% of the spawning cells for each species, there were more differences in the coefficients for the regression model, but the overall role of each trait remained unchanged (Table 5.2). The direction of the impact for DVM and surface-tracking behaviors remained unchanged, but pycnocline seeking behavior exhibited a non-significant negative effect under the new regression instead of the non-significant positive effect under the original. Overall, the magnitude of the effect of vertical swimming behavior was reduced. The direction of the impacts for spawning substrate, minimum PLD length, and the spawning time also remained unchanged, but the magnitude differed from the original. Finally, the impact of the maximum settlement depth and variance in the settlement probabilities did not change. Overall, these regressions indicate that our results are not sensitive to northward species range contractions. However, as we discuss further in section 5.4, the issue of predicting the impacts of climate change on larval dispersal is substantially more complex than we have portrayed it in out analysis.

5.4 Discussion

The primary objective of this study was to identify the traits and combinations of traits that are most influential in determining dispersal success and spatial structure. We identified a number of traits that influence dispersal success rates, but did not observe any statistically significant interaction terms between the traits. The clustering

algorithm from chapter 4 also identified a number of regimes that highlight the role of specific traits in determining the spatial patterns of dispersal. These results provide insight into the relationships between species traits and larval dispersal patterns, and the framework used here could be used to explore other study systems or to evaluate how the Gulf of Maine system may respond to different a forcing regime.

Of the traits that influence the spawning distribution, the spawning substrate had the strongest influence on dispersal success, the mean spawning time had a lesser impact, and the maximum spawning depth did not significantly influence the results. This result is largely driven by a strong correlation between the location of gravel substrate and shallow regions, also highlights important spawning areas for species in the Gulf of Maine. Species that spawned exclusively over gravel were associated with a significantly higher proportion of successfully dispersing larvae, and the gravel substrate is primarily found in the Great South Channel, on Georges Bank, and along the southern and western sides of Nova Scotia. At least two of these regions, the Great South Channel and Georges Bank, are common spawning areas for a variety of species, and our results help to provide insight into why species may preferentially spawn here. Because many species spawn in these areas, a number of studies have examined the processes that lead to successful retention and larval dispersal here, and many of these studies are highlighted in section 5.1. The second area that consisted of largely gravel habitat was along the coast of Nova Scotia, and our analysis of Lagrangian coherent structures in Appendix D suggests that there is a persistent barrier to transport at the mouth of the Bay of Fundy. Overall, our result that species who spawn exclusively over gravel have higher dispersal success highlights important spawning areas in the region.

After spawning, we found that the minimum PLD length and vertical swimming behavior were the most important traits for determining dispersal success. As we predicted, species with shorter PLDs were significantly more likely to successfully disperse than those with longer PLDs. Logically, this result makes sense because an important aspect of successful dispersal is retention within the vicinity of suitable settlement habitat. Notably, the duration of the competency window did not emerge as a significant trait in determining dispersal success. This result suggests that once a larva exits the vicinity of suitable habitat, it is unlikely to return. Considering that the Gulf of Maine contains recirculating flow patterns, but is bounded by southward moving continental slope and shelf waters that would swiftly sweep larvae away (Townsend et al., 2004), this result is unsurprising. Our simulation also predicted that surface-tracking behavior significantly decreased the proportion of larvae that successfully settled relative to passive particles, presumably because faster moving surface currents are more likely to transport larvae away from the study region. We also found that pycnocline-seeking behavior non-significantly increased the proportion of larvae that successful dispersal, which did not agree with our initial hypothesis that active swimming behaviors, including pycnocline-seeking and DVM, would increase retention for species with long PLDs. Although prior studies often indicated that swimming behaviors may increase larval retention, those same studies also indicated that the results may vary geographically and based on the current oceanographic conditions (Gilbert et al., 2010; Churchill et al., 2011). As with some of

the processes that we chose not to include in our model, notably growth and mortality, swimming behaviors vary widely between species. It is likely true that swimming behaviors for each species are adapted to work well with the dietary demands, predation pressure, and specific attributes of that species, so including it in a generalized model such as ours is unlikely fully capture the complexity of the process. Nonetheless, the strong effect of surface-tracking as opposed to the other behaviors helps to shed insight into the importance of dispersal depth overall, and further work into parameterizing and simulating complex behaviors may be warranted.

As expected, we found that that traits which increase the available settlement habitat for a species also increase the dispersal success rate. We modeled suitable settlement habitat based on the substrate composition and bathymetry, so species with deep maximum settlement depths and minimal substrate preference resulted in the highest levels of dispersal success. These traits, together with the maximum spawning depth that determines the availability of spawning habitat, were also the primary traits that determined the spatial clustering patterns for each species. These results indicate that the location and abundance of spawning and settlement habitat has a strong impact on dispersal success rates and population spatial structure. For many species, suitable habitat is also characterized by other variables, particularly water temperature (Pearce et al., 2004; Simpson and Walsh, 2004; Nye et al., 2009), and changes to these variables may drive changes to both the rates of dispersal success and spatial structure in the region. We conducted a simple sensitivity analysis that explores the possibility of a northward range contraction in species distribution patterns, and found that it had a fairly minor impact on the dispersal success rates. Although poleward distribution shifts have been observed for many species in multiple geographic regions and it is hypothesized that they will continue into the future (Perry et al., 2005; Nye et al., 2009; Kleisner et al., 2017), simply shrinking the size of the study domain by cutting off the southern end may not be sufficient to adequately capture the full complexity of the issue. For many of the species that have shifted north, it is believed that the underlying cause of species distribution shifts is the interaction between warming global oceans and static themal tolerances (Perry et al., 2005; Nye et al., 2009; Kleisner et al., 2017). Thermal influences have also been observed to result in movement to deeper waters (Perry et al., 2005), and the interactions between bathymetry and latitude may result in distributions shifting in other directions than only poleward. A more accurate way to assess the impact of climate change on larval dispersal would be to include thermal tolerances for each species or add other indicators of habitat quality and directly model how these traits influence dispersal patterns. Although this work is beyond the scope of this study, it would be a useful extension.

In addition to directly impacting species distribution patterns, changes to the physical environment are likely to have indirect effects through other processes. Both the intensity and direction of physical processes naturally vary in time, and changes to the environmental properties that regulate species distributions are likely to impact these processes as well. Churchill et al. (2011) demonstrated that the retention of cod larvae on the New England coast is substantially impacted by the presence of upwelling vs. downwelling regimes, and Tian et al. (2009b) related the retention

of Georges Bank scallop larvae to the strength of the tidal mixing front. Further analysis of the robustness of our results to variability in these processes and other forcing properties would be highly informative, and may be possible using available forcing data. The FVCOM forcing that we used to drive our IBM has been generated for 1978 through the present, and our simulation could theoretically have been run for multiple years that represent different forcing regimes. The results of that exercise would provide insight into the relationships between larval success, spatial patterns of dispersal, and physical processes, and from those results it may be possible to infer how the results may change in response to future focing regimes. Although running multiple years of simulations at the scale of this study is not computationally practical with our current resources, the rapid development of new technology will likely make it possible within the near future.

Although this study provides valuable insight into how species traits influence dispersal patterns in the Gulf of Maine, we would caution against using the results of this study alone to steer management decisions. Our model is necessarily a simplification of reality and excluded a number of processes, including some potentially important ones. The most prominent processes excluded from our model were larval growth, mortality, and the processes governing the juvenile and adult stages. Some of the processes that we included, such as the DVM and pycnocline-seeking behaviors, are likely to be driven by responses to food availability and predation. As a result, notwithstanding our observations that they did not significantly influence dispersal success and the population connectivity patterns, they may do so indirectly by influencing larval survival. For example, Petrik et al. (2014) simulated larval haddock dispersal using an IBM that includes bioenergetics and found that DVM actually decreased larval survival. Much of this study focused on examining the spatial patterns of dispersal during the larval stage, but adult movement may also influence the results. For instance, Atlantic herring are highly migratory and adult movement patterns may substantially alter the connectivity patterns. Explicitly modeling these processes may appreciably change our results, but also has the potential to result in a model with unmanageable complexity. Finally, population connectivity patterns are an emergent property of the ecosystem that result from the interaction between many physical, chemical, and biological factors. Although the TBM framework is an effective technique for systematically assessing the role of many different factors across a broad range of potential species, it does so by excluding potentially important details and should be complemented with species specific studies that faithfully represent these details.

Overall, this study identified a set of traits that regulate the dispersal success and patterns for species in the Gulf of Maine, and these traits align well with the results of other TBM studies. In contrast to prior studies that examined a single or few species in detail, this study explored a variety of species that exhibit vastly different reproductive strategies. We are aware of only one other published TBM study for marine larval dispersal, and it arrived at similar conclusions to ours. Treml et al. (2015) examined the dispersal of reef fish in the mostly enclosed Port Phillip Bay, Australia, and they concluded that larval mortality, PLD duration, and the traits that regulate spawning and settlement habitat requirements were the most influential

for determining population connectivity patterns. Although we did not include larval mortality, we came to the same conclusion about the other traits. Moving forward, we hope to see the TBM framework applied to a variety of other study systems and forcing regimes, and we would include larval mortality in future work. We expect to see that the traits identified by Treml et al. (2015) and by this study will emerge as the primary drivers of larval dispersal patterns across a wide range of study systems, and that information would help steer complementary observational and modeling work into promising areas of research.

Chapter 6

Concluding Remarks

Larval dispersal and population connectivity may have important effects on the demographics, spatial structure, and choice of management protocols for marine ecosystems. Individual-based models (IBMs) provide complementary data to field observations and assist in developing a comprehensive understanding of the mechanisms and patterns of larval dispersal. This thesis presents 3 new technologies for IBM studies and an application of these technologies to modeling larval dispersal in the Gulf of Maine.

In chapter 2, we present a novel procedure for identifying the minimum number of particles required to achieve statistical convergence in IBM studies. This procedure both identifies when statistical convergence has been achieved and allocates particles among multiple release sites to minimize the computational requirements. Using this procedure, we achieved convergence after simulating 959 million particles in chapter 5. Had we used only the statistical model, but not the optimization heuristic, from our procedure, it would have required 3.3 billion particles to achieve the same level of statistical confidence. The absence of our statistical model would have required multiple replicate simulations to assess the robustness of our results, and the study would have been beyond our current computational capabilities.

Chapter 3 presents a second, complementary attempt to reduce the computational expense of IBM studies. In that chapter, we present an IBM that was capable of running on either central processing units (CPUs) or graphics processing units (GPUs). Ultimately, we found that some tasks execute more efficiently on CPUs, and others on GPUs. As a result, models that utilize both computing architectures are likely to be the most efficient users of computational resources. Although we ultimately ran our IBM simulations on CPUs due to pragmatic concerns about the computing resources available to us, the exploration of GPU-based IBMs provides valuable insight into IBM design. However, as we describe in chapter 3, additional optimization of the data structures and development of algorithms that run efficiently on GPUs would likely further improve the performance of IBMs.

In chapter 4, we present a clustering algorithm that seeks to identify coherent geographic regions from population connectivity data while simultaneously considering multiple species. We compare the results of this algorithm with results from two existing algorithms for processing single species data. We find that all three algorithms

identify similar clustering patterns, and that the new algorithm effectively identifies similar species and maintains the geographic clustering patterns for each species when averaging over them. Applying the multigraph algorithm from this chapter to the trait-based model in chapter 5, we find that it provides useful insight into the role of individual traits in determining connectivity patterns.

Finally, in chapter 5, we apply a trait-based modeling (TBM) approach to simulating larval dispersal in the Gulf of Maine. Under this approach, species are represented by a generalized model that includes only the most important traits. We simulate a wide variety of trait combinations using this model, and find that the geographic distribution of spawning and habitat requirements for settlement are the most important for determining dispersal success and patterns.

Although this thesis provides valuable insight into the Gulf of Maine ecosystem and presents new tools for individual-based modeling in general, additional research could enhance all of the chapters. Chapters 2 and 4 both present novel algorithms for addressing optimization problems. Although the algorithms presented in both chapters appear to work well for the study systems on which they were tested, the development of additional theory would strengthen the support for both. In particular, it would be helpful to provide a theoretical upper bound on the limit between the optimal solution to the particle allocation and clustering problems and the suboptimal solutions that are provided by our optimization heuristics. For chapter 3, additional optimization of the model implementation may be useful. Although we attempted to reasonably improve the model performance, additional performance testing and refinements to the data structures and algorithm choices would likely result in reduced runtime. These efforts could result in a highly efficient core codebase that implements the particle-tracking, but allows for biological processes to be layered on top easily. Finally, chapter 5 provides the greatest possibility to enhancement. The coupled biological and physical processes that shape population connectivity patterns are incredibly complex, and our model only includes a very limited subset of them. Both adding additional complexity to our model and applying it to different regions would increase the robustness and utility of the predictions from it.

Overall, this thesis presents a collection of algorithms, software, and simulation results that assist in understanding marine larval dispersal processes. We hope that these tools and insights are useful to other researchers in the field and that they can be built upon in the future.

Appendix A

Sequential Analysis Pseudocode

This appendix provides pseudocode for a naive implementation of the allocation rule. We recommend that readers refer to the software packages referenced in the main text for more computationally efficient implementations.

function COMPUTEEXPECTEDPOSTERIORCOST($\alpha_i^{(k)}$, n)
  while estimate for $H^{(k+1)}$ not converged **do**
    $p_i^* \leftarrow$ draw from a Dirichlet distributions with parameters $\alpha_i^{(k)}$
    $d \leftarrow$ draw n particles from a multinomial distribution with parameters p_i^*
    $\alpha_i^{(k+1)} \leftarrow \alpha_i^{(k)} + d$
    estimate $H^{(k+1)}$
  end while
  return estimated $H^{(k+1)}$
end function
$b \leftarrow$ number of particles to allocate
$m \leftarrow (0, 0, \ldots, 0)$ (release distribution)
for $1, 2, \ldots, b$ **do**
  for i in origins **do**
    $H_i^{(k+1)} \leftarrow$ computeExpectedPosteriorCost($\alpha_i^{(k)}$, $m_i + 1$)
  end for
  $i \leftarrow \arg\min(H_i^{(k+1)})$
  $m_i = m_i + 1$
end for

Appendix B

Sequential Analysis Demonstration

In section 2.4, we described the application of our method to a simulation of the Gulf of Maine. This demonstration traces the details of that process for the first replicate simulation and may serve as a template for other researchers who seek to apply our method. After the description of each step, we provide R code that implements it using our package.

We begin by specifying 3 origin regions (1, 2, & 3), and 5 destination regions (Mid-Maine, Three States, Mass Bay, Nantucket, & Other). The objective of our simulation is to estimate the matrix P so that the value of the objective function defined in Equation 6 is less than 0.1 ($\epsilon = 0.1$) and excluding any p_{ij} for which we are at least 95% confident that $p_{ij} \leq 0.005$ ($\delta = 0.005$, $\pi = 0.05$). We specify that we would simulate batches of 500 particles, up to a maximum of 50,000 particles total based upon the estimated computer time to complete each simulation and the computational resources available to us.

```
# Load the library.
library(jsj2016)
# Set the basic simulation parameters.
origins = as.factor(c(1, 2, 3))
destinations = c('Mid-Maine', 'Three States', 'Mass Bay', 'Nantucket', '
    Other')
batch_size = 500 # particles.
budget = 50000 # particles.
# Save the objective function and the parameter list for it.
obj_fn = jsj2016::obj_fn_probabilities # Objective function
obj_fn_arg = list(epsilon = 0.1, delta = 0.005, pi = 0.05)
# Create the initial prior vectors and save them as a matrix.
alpha = matrix(1, length(origins), length(destinations))
# Create the initial count vectors and same them as a matrix.
x = matrix(0, length(origins), length(destinations))
alphas = 1 + x
```

For the first step, we have no prior knowledge of the connectivity matrix that we will estimate. Therefore, we uniformly allocate particles across the 3 origin sites and create the initial priors, $\alpha_i^0 = (1, 1, 1, 1, 1), i = 1, 2, 3$. After simulating the first

1,000 particles, we computed the vectors, $x_i^{(1)}, i = 1, 2, 3$, which give the number of particles released from origin i that arrived at destination j in this first simulation. In this case, $x_1^{(1)} = (5, 3, 15, 2, 142)$, $x_2^{(1)} = (0, 1, 30, 1, 135)$, and $x_3^{(1)} = (0, 0, 57, 2, 107)$. Using these results, we update our prior vectors to $\alpha_1^{(1)} = 1 + x_1^{(1)} = (6, 4, 16, 3, 143)$, $\alpha_2^{(1)} = 1 + x_2^{(1)} = (1, 2, 31, 2, 136)$, and $\alpha_3^{(1)} = 1 + x_3^{(1)} = (1, 1, 58, 3, 108)$.

```
# Evaluate the objective function.
if(obj_fn(alpha, obj_fn_arg) < obj_fn_arg$epsilon) {
  print('Simulation completed successfully.')
} else {
  print('Continue simulation')
}
# Assess if we can afford more particles.
if(sum(x) >= budget) {
  print('Simulation failed, budget exceeded.')
} else {
  print('Continue simulation.')
}
# Allocate the particles.
release_distribution = optimization_heuristic(alpha, batch_size, obj_fn,
    obj_fn_arg)
print('Release distribution:')
print(release_distribution$dist)
# Simulate the next batch of particles using an individual based model.
# Update x and alpha.
x[1,] = x[1,] + c(5, 3, 15, 2, 142)
x[2,] = x[2,] + c(0, 1, 30, 1, 135)
x[3,] = x[3,] + c(0, 0, 57, 2, 107)
alpha = 1 + x
```

To begin step 2, we first assess if more particles need to be simulated. We compute the value of the objective function defined by Equations 3-6, $H^{(2)} = 0.994$. $H^{(2)}$ is greater than our threshold value of 0.1, so a second simulation is required. We are still within our computational budget of 50,000 particles, so we proceed to allocate the particles among origin regions. Our allocation scheme suggests releasing 43 particles from origin 1, 228 particles from origin 2, and 229 particles from origin 3. Using a particle-tracking model, we simulate this batch of particles. Observing that of the 43 particles released from origin 1, 2 went to Mid-Maine, 1 to Three States, 4 to Mass Bay, and 0 to Nantucket, $x_1^{(2)} = x_1^{(1)} + (2, 1, 4, 0, 36) = (7, 4, 19, 2, 178)$ and $\alpha_1^{(2)} = 1 + x_1^{(2)} = (8, 5, 20, 3, 179)$. The count and parameter vectors for origins 2 and 3 are updated in an identical fashion.

```
# Evaluate the objective function.
H = obj_fn(alpha, obj_fn_arg)
if(H < obj_fn_arg$epsilon) {
  print('Simulation completed successfully.')
} else {
  print(sprintf('Continue simulation, H = %f', H))
}
# Assess if we can afford more particles.
```

```
if(sum(x) >= budget) {
  print('Simulation failed, budget exceeded.')
} else {
  print('Continue simulation.')
}
# Allocate the particles.
release_distribution = optimization_heuristic(alpha, batch_size, obj_fn,
    obj_fn_arg)
print('Release distribution:')
print(release_distribution$dist)
# Simulate the next batch of particles using an individual based model.
   Then repeat.
# Update x and alpha.
x[1,] = x[1,] + c(2, 1, 4, 0, 36)
x[2,] = x[2,] + c(0, 4, 45, 1, 178)
x[3,] = x[3,] + c(0, 0, 85, 1, 143)
alpha = 1 + x
```

We repeat the same process for steps 3, 4, $\cdots$, 53. At the end of step 53, 26,500 particles have been simulated. Using $\alpha_1^{(53)} = (326, 121, 1644, 103, 15447)$, $\alpha_2^{(53)} = (2, 101, 1162, 27, 4582)$, and $\alpha_3^{(53)} = (1, 1, 1060, 15, 1923)$ to compute $H^{(54)}$ at the beginning of the next step, we find that $H^{(54)}$ is below our threshold of 0.1, and so we conclude the simulation with a successful result.

```
# Update alpha.
alpha[1,] = c(326, 121, 1644, 103, 15447)
alpha[2,] = c(2, 101, 1162, 27, 4582)
alpha[3,] = c(1, 1, 1060, 15, 1923)
# Evaluate the objective function.
if(obj_fn(alpha, obj_fn_arg) < obj_fn_arg$epsilon) {
  print('Simulation completed successfully.')
} else {
  print('Continue simulation')
}
```

Appendix C

Example IBM Configuration File

This appendix presents an example configuration file for the individual-based model presented in chapter 3 and used throughout the dissertation. Each option is documented immediately before it appears. The configuration is stored in JSON format. The text offset in C style comments (e.g. /* comment */) is provided for illustation purposes only.

```
{
    /* The number of OpenMP threads or CUDA threads per block. */
    "n_threads": 1,
    /* One or more FVCOMDataset objects may be included. Each one
     * corresponds to an instance of the FVCOMDataset class and should
     * be given a unique name. */
    "FVCOMDataset": {
        /* The name of this dataset. */
        "name": "fvcom_gom3",
        /* The name of the file containing the FVCOM mesh. */
        "mesh_filename": "/scratch/gom_hourly/gom3_199501.nc",
        /* The name of the file containing the bottom substrate data and
         * mixed layer depth. */
        "extra_filename": "/scratch/data/extra_data.nc",
        /* An array of filenames for the forcing data. */
        "filenames":
            ["/scratch/gom_hourly/gom3_199501.nc",
            "/scratch/gom_hourly/gom3_199502.nc"]
    /* One or more ParticleGroup2d, ParticleGroup3d, or LarvaGroup
     * objects may be included. Each one will result in a single
     * instance of the corresponding class in our model. */
    }, "LarvaGroup": {
        /* The name of the input file containing the release coordinates
         * and time for each particle. */
        "input_filename": "init_000.nc",
        /* The name of this LarvaGroup instance. */
```

```
"name": "igroup",
/* The frequency at which the particle states should be written
 * to the output file (e.g. every 144th timestep).
"output_frequency": 144,
/* The timestep to use for particle trajectory integration
 * (units: seconds). */
"timestep": 600,
/* The name of the forcing dataset to use. A FVCOMDataset object
 * with this name must also appear in this file. */
"forcing_dataset": "fvcom_gom3",
/* The name of the output file. */
"output_filename": "out_001.nc",
/* The end time for this run (units: modified Julian date). */
"end_time": 50024.230000,
/* The settlement probability on fine sand. This option is only
 * applicable to the LarvaGroup class and will be ignored by
 * the ParticleGroup2d and ParticleGroup3d classes. */
"settlement_prob_fine_sand": 0.476191,
/* The settlement probability on coarse sand. This option is only
 * applicable to the LarvaGroup class and will be ignored by
 * the ParticleGroup2d and ParticleGroup3d classes. */
"settlement_prob_coarse_sand": 0.476191,
/* The settlement probability on gravel. This option is only
 * applicable to the LarvaGroup class and will be ignored by
 * the ParticleGroup2d and ParticleGroup3d classes. */
"settlement_prob_gravel": 0.047619,
/* The minimum PLD before settlement. This option is only
 * applicable to the LarvaGroup class and will be ignored by
 * the ParticleGroup2d and ParticleGroup3d classes (units: days).
 */
"min_pld": 55.000000,
/* The duration of the competency window. This option is only
 * applicable to the LarvaGroup class and will be ignored by
 * the ParticleGroup2d and ParticleGroup3d classes (units: days).
 */
"competency_window": 20.000000,
/* The maximum settlement depth. This option is only
 * applicable to the LarvaGroup class and will be ignored by
 * the ParticleGroup2d and ParticleGroup3d classes (units: m).
 */
"settlement_max_depth": -100.000000,
/* Should this species engage in diel vertical migration?
 * This option is only applicable to the LarvaGroup class and
 * will be ignored by the ParticleGroup2d and ParticleGroup3d
 * classes. */
```

```
        "dvm": false,
        /* The maximum swimming speed for this species. This option is
         * only applicable to the LarvaGroup class and will be ignored
         * by the ParticleGroup2d and ParticleGroup3d classes
         * (units: m / s). */
        "max_swim_speed": 0.000000,
        /* Should this species remain fixed at the release depth?
         * This option is only applicable to the LarvaGroup class and
         * will be ignored by the ParticleGroup2d and ParticleGroup3d
         * classes. */
        "fixed-depth": false,
        /* Should this species swim towards the mixed layer depth
         * specified in the extra file? This option is only applicable
         *  to the LarvaGroup class and  will be ignored by the
         * ParticleGroup2d and ParticleGroup3d classes. */
        "variable-target-depth": false
    }
}
```

Appendix D

Lagrangian coherent structures in the Gulf of Maine

Lagrangian Coherent Structures Lagrangian coherent structures (LCS) may be defined as material surfaces in fluids that move with the flow and organize the flow into ordered patterns (Haller, 2015). LCS analyses have recently been applied to a variety of problems in oceanography including characterizing larval transport patterns (Harrison et al., 2013). This appendix describes our analysis of LCS in the Gulf of Maine as it applies to larval transport.

Finite-time Lyapunov exponents (FTLEs) are a common method to numerically identify LCS (Shadden et al., 2005). FTLEs quantify the stretching and deformation of a unit volume of fluid as it is advected. Computing the value of the FTLE at any point (x_0, y_0, t_0) is a three step process that involves first particle-tracking, then approximating the strain tensor from the particle trajectories, and finally computing the FTLE value from the strain tensor. Consider 4 particles that are released at the points ($x_0 + \Delta x$, y_0, t_0), ($x_0 - \Delta x$, y_0, t_0), (x_0, $y_0 - \Delta y$, t_0), and (x_0, $y_0 + \Delta y$, t_0). After advecting these particles for T days, the new positions are (x_1, y_1, $t_0 + T$), (x_2, y_2, $t_0 + T$), (x_3, y_3, $t_0 + T$), and (x_4, y_4, $t_0 + T$). The strain tensor, A, may then be approximated by Equation D.1.

$$A = \begin{pmatrix} \frac{x_2 - x_1}{2\Delta x} & \frac{x_4 - x_3}{2\Delta y} \\ \frac{y_2 - y_1}{2\Delta x} & \frac{y_4 - y_3}{2\Delta y} \end{pmatrix} \tag{D.1}$$

The FTLE is then given by Equation D.2.

$$\text{FTLE} = \log \frac{\lambda_{\max}(A'A)}{T} \tag{D.2}$$

When T is specified to be a positive time, the forward-time FTLE (FFTLE) results from these calculations, and conversely, the reverse time FTLE (RFTLE) is computed when T is a negative time (i.e. particle-tracking backwards in time). Computing the FTLE values at a grid of spatial locations results in a map, and ridges of strongly positive values in this map indicate the location of LCS. Ridges of FFTLEs indicate the boundary between divergent regions in the flow field, and ridges of RFTLEs

indicate curves towards which particles will be attracted.

Particle-tracking We computed daily FTLE field snapshots for the Gulf of Maine during 1995. Particles were released daily at midnight on a 1 km grid at 1 m depth, then tracked for up to 20 days using a 10 minute timestep. The particle-tracking was completed using FISCM, which integrates trajectories with a modified 4^{th}-order Runge-Kutta timestep. Neither horizontal nor vertical diffusion was included. FISCM was forced using the hourly archived output from FVCOM that is described in detail in chapter 5.

We then computed daily snapshots of the FTLE field are various choices of T and using $\Delta x = \Delta y = 500\ m$. To better understand how the LCS patterns may vary over the course of the year, we applied a form of principal components analysis. We reshaped each FTLE field into a column vector, then stacked the column vectors to form a matrix with 1 column for each particle release time and 1 row for each spatial location. After computing a singular value decomposition (SVD) of this matrix, we reconstructed the FTLE fields using only the minimum number of singular values required to represent 50% of the variance in the original data.

Results The distribution of FTLE values is strongly influenced by the choice of T, and we explored a variety of possibilities for T (Figure D-1). After only 1 day, the FTLE field reveals only the strongest LCS (e.g. surrounding Georges Bank), and is dominated by small values. Two days later, these LCS continue to strengthen and new LCS appear along the southern edge of Browns Bank and at the mouth of the Bay of Fundy. After 1 week of integration, these LCS become stronger and a tangled mess of LCS appear in the nearshore areas and along the shelf break. Longer integrations times (e.g. 2 weeks) strengthen the patterns from the 1 week integration.

The SVD based reconstruction of the FTLE fields highlighted regions that consistently exhibited LCS Figure D-2. For both the 2D and 3D integration, the most extreme values of the RFTLEs and FFTLEs are located along the shelf break, at the northern edge of Georges Bank, and at the mouth of the Bay of Fundy. Theses values are more extreme for the 3D than 2D advection case. Many of the northern areas also contain moderately strong FTLE values.

Discussion Although the LCS analysis highlighted some regions that commonly contain strong LCS, it suffers from a fatal flaw that renders it unsuitable for inclusion in our analysis of larval dispersal. Comparison of the locations of extreme FTLE values with the FVCOM mesh revealed that the strong FTLE values occur near small FVCOM mesh elements. This correlation makes intuitive sense because complex circulation patterns with many small scale features are likely to result in large FTLE values, and small mesh elements are better able to capture small scale complexity in the flow regime than larger elements. However, these small mesh elements were deliberately placed in regions where the real-world flow is expected to be complex, and it is impossible to differentiate between LCS complexity due to real-world flow complexity and LCS complexity due to the mesh structure in our model configura-

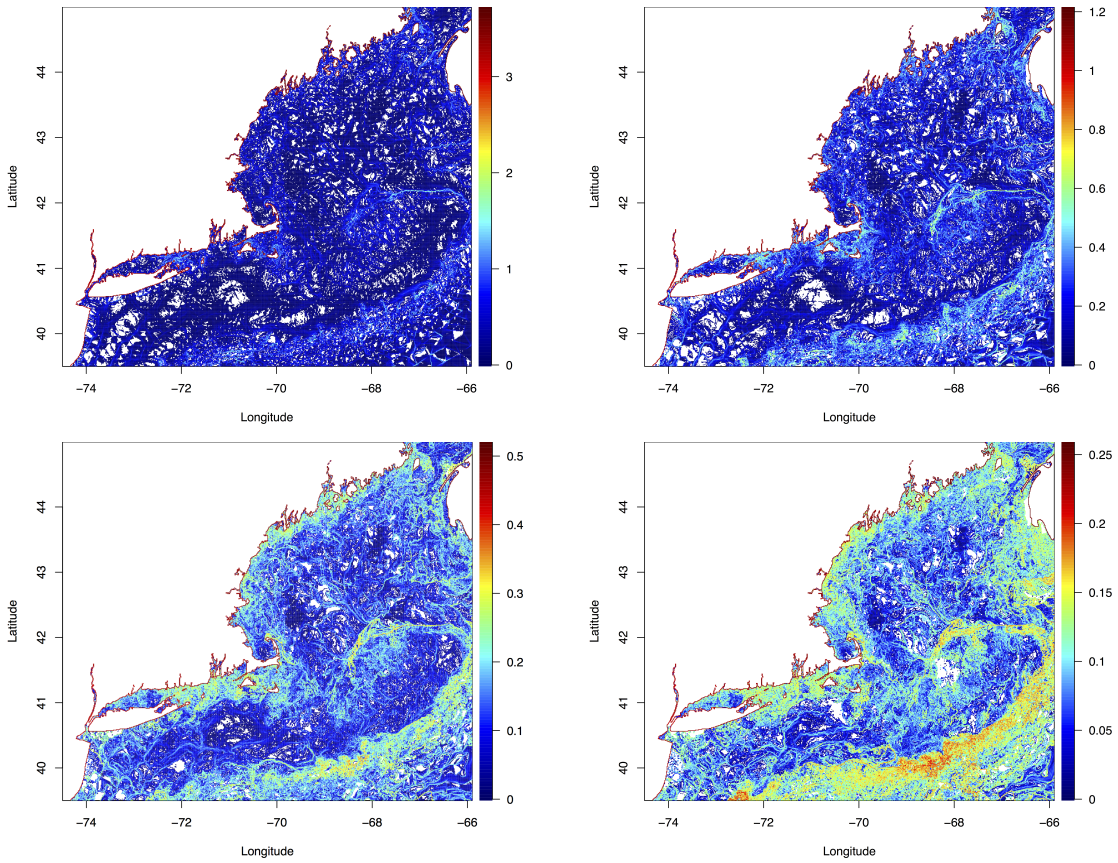

Figure D-1: The values of the FFTLE field are plotted for multiple integration times and for a release time of 1 Jan 1995. Clockwise from top left, the particle trajectories were integrated for 1 day, 3 days, 14 days, and 7 days before computing the strain tensor.

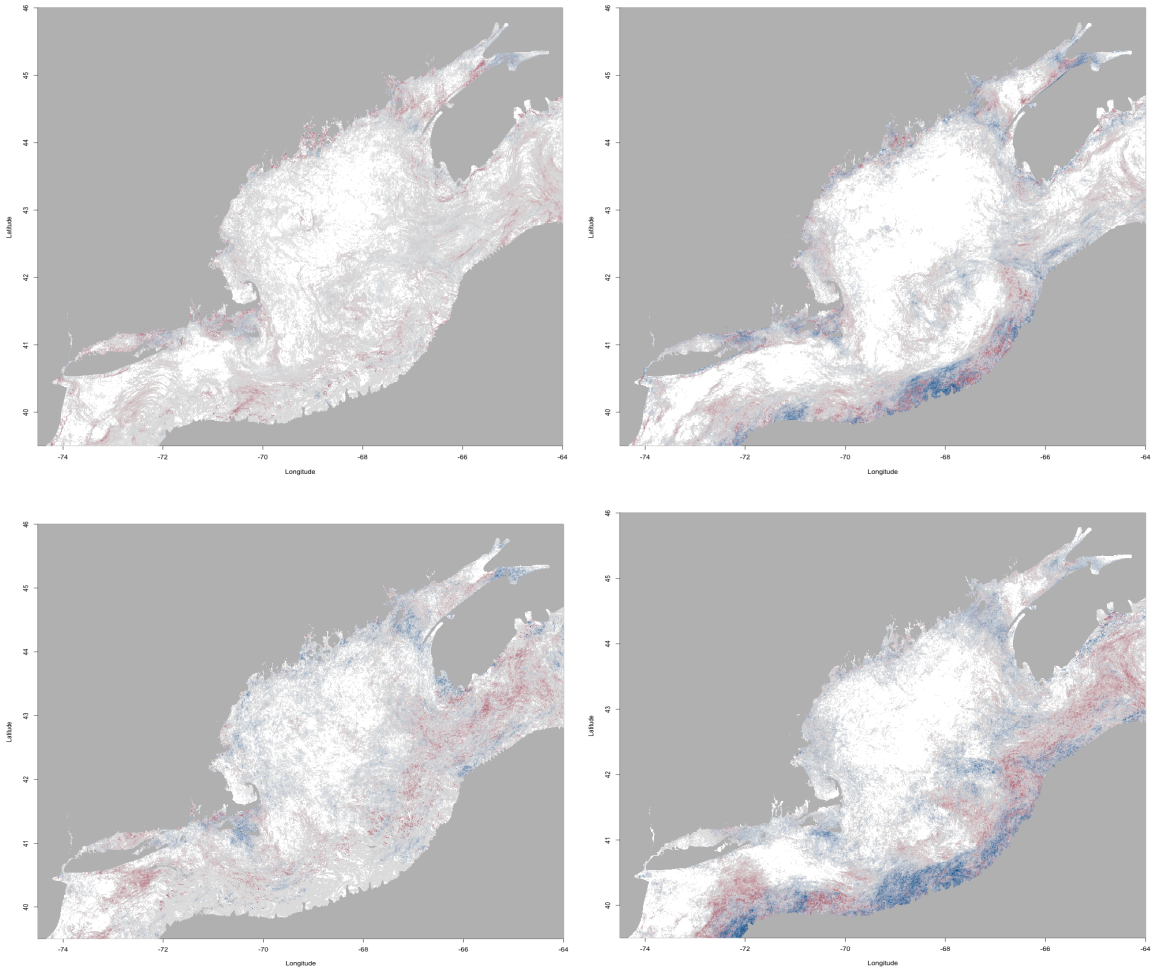

Figure D-2: The reconstructed FTLE fields are plotted for the 7 day integration period (top row) and 20 day integration time (bottom row). The left column depicts FTLEs that were computed from particles integrated in 2D at 1 m depth, and the bottom row depicts FTLEs from 3D integration. Red regions indicate strongly positive FFTLEs and blue regions indicate strongly positive RFTLEs. The grey areas indicate more moderate FTLE values, and the white areas indicate FTLE values near 0.

tion. Prior to assessing larval dispersal patterns using LCS and a variable resolution hydrodynamic model, we would suggest that additional research into the relationship between mesh resolution and LCS calculations be explored.

Appendix E

Species Trait Parameterization

This appendix provides details about how the individual-based model from chapter 5 was parameterized. It describes the both process for selecting the 4 real world species and the distributions from which each parameter value was drawn for the artificial species.

The analysis is based on the life history for 24 species that live in the Gulf of Maine and surrounding areas. These species include Atlantic cod (AC), haddock (HD), pollock (PO), silver hake (SH), red hake (RH), white hake (WH), yellowtail flounder (YF), winter flounder (WT), American plaice (AP), witch flounder (WF), ocean pout (OP), goosefish (GO), Acadian redfish (AR), little skate (LS), winter skate (WS), thorny skate (TS), barndoor skate (BS), smoth skate (SS), Atlantic mackerel (AM), butterfish (BF), Atlantic herring (AH), spiny dogfish (DF), ocean quahog (OQ), and sea scallop (SS). For the most part, the parameter values for each species were taken from the Essential Fish Habitat Technical Memorandums that are published by the NOAA Northeast Fisheries Science Center. The full list of parameter values for each of these species and the citations for their sources are reported in Table E.1, Table E.2, Table E.3, and Table E.4.

E.1 Variables

We considered 3 taxonomic and 16 other variables for each species. This section defines each variable and describes how it was estimated. In many cases, the variables were not reported in the existing literature and were inferred from other variables.

Taxonomic variables Each species was identfied by its common name, latin name, and the grouping given in Liu et al. (2012).

Adult movement potential The adult movement potential describes the level of geographic movement for adults of each species. It is a categorical variable that takes one of three values: low, medium, or high. Species with low adult movement potential are those that do not undertake seasonal or other migrations. Species with medium

potential are those that undertake some seasonal migrations (e.g. nearshore to offshore), but have limited mixing on broad scales. Species with high adult movement potential are those that have demonstrated high levels of movement and are mostly pelagic species such as Atlantic herring.

Egg size The egg size is reported as the mean of the minimum and maximum reported egg diameter in μm. For species that give live birth (e.g. dogfish), the egg size was recorded as the size at birth.

Larval depth The larval depth is recorded as the minimum and maximum depths at which larvae are usually found for each species.

Longevity The longevity is the maximum age for the typical adult female in years.

Maturation age and length The maturation age (A_{50}) and maturation length (L_{50}) are the age in years and size in mm at which the 50^{th} percentile of females mature. Multiple estimates were often available for each species. When available, the A_{50} and L_{50} estimates from the NOAA NEFSC surveys were used, and estimates local to Georges Bank or the Gulf of Maine were preferentially chosen.

Maximum clutch size The maximum clutch size is the number of eggs that may be produced in a single year by the largest females of the species. This variable was not available for all species, and leaving it blank would result in the species being eliminated from the PCA. To avoid the exclusion of some species, it was filled with a value from a similar species when an estimate of fecundity was not available.

Maximum length The maximum length in the length of the largest observed individual in in mm. The maximum observed value is unlikely to be typical of individuals within the species, so it was not used for the principal components analysis (PCA) described later.

Pelagic larval duration limits The pelagic larval duration (PLD) is defined by the minimum and maximum length in days. This value describes the number of days that each species is expected to spend within the water column before recruiting as a juvenile. For many species, the PLD limits were not directly reported in existing literature and were instead calculated from the larval growth rate and size at maturity.

Spawning dates The start of the spawning season and end of the spawning season were identified for each species. In addition, a spawning season duration was computed as the number of days during the year during which a species usually spawns. There is a high level of uncertainty associated with the spawning season start, end, and duration variables.

E.2 Principal Components Analysis

We used principal components analysis (PCA) to choose a few species that represent the diversity of life history strategies in the region. The analysis focused on the adult and reproductive portions of the life cycle, but excluded most larval traits. As a result, it included only the duration of the primary spawning season, maximum clutch size, longevity, maturation age, egg size, maturation length, and adult movement potential. All variables were rescaled to have mean 0 and unit variance prior to applying the PCA procedure.

Our first iteration of PCA identified ocean quahogs, dogfish, and the skates as the most distinct species (Figure E-1). Ocean quahogs are an outlier species due to their longevity. Dogfish and skates formed a separate cluster because they give live birth (dogfish) or lay exceptionally large eggs (skates). However, none of these species are good candidates for individual-based modeling. The parameter values for ocean quahogs are poorly known, skate eggs tumble on the bottom, and dogfish give live birth, so simulating the larval stage of these species is not feasible. We therefore excluded these species from the remainder of the analysis.

A second iteration of PCA excluded the species noted above and highlighted a few candidates for individual-based modeling (Figure E-2). The first principal component captured 36% of the variance and positive values of this component described migratory, rapidly maturing, small species with small eggs. This component is similar to the second principal component in Winemiller and Rose (1992), Table 5. The second principal component captured an additional 25% of the variability in the data, and positive values of this component described species that mature at large sizes, are somewhat migratory, and spawn few eggs over a prolonged spawning season. Based on the results of the PCA, we modeled sea scallops, haddock, Atlantic herring, and yellowtail flounder in chapter 4 and chapter 5. These species capture a variety of life history strategies and have been well enough studied to parameterize an individual-based model.

E.3 Distributions

Based on the parameter values reported in section E.4, we used the following distributions to generate artificial species for chapter 5.

Spawning time The mean was generated by sampling from a uniform distribution across the entire year. The standard deviation was generated by drawing from a lognormal distribution with mean 21 days and a standard deviation of $e^{0.75}$ days.

Spawning substrate The spawning substrate was chosen by randomly drawing one of sand, gravel, or both.

Maximum spawning depth The maximum spawning depth was chosen by sampling from a lognormal distribution with mean $e^{4.5}$ m and standard deviation $e^{0.5}$ m.

Behavior The vertical swimming behavior was chosen by randomly choosing either passive particles, surface-tracking, pycnocline-seeking, or diel vertical migration.

Minimum PLD The minimum PLD was generated by sampling from a lognormal distribution with mean 60 days and standard deviation $e^{0.25}$ days.

Competency Window The competency window was generated by sampling from a normal distribution with mean equal to 20% of the minimum PLD for the species and a standard deviation of 2 days.

Settlement substrate The probability of settling on each of gravel, coarse sand, and fine sand was generated by sampling from a Dirichlet distribution with parameter vector $(1, 1, 1)$.

Maximum settlement depth The maximum settlement depth was chosen by sampling from a lognormal distribution with mean $e^{4.5}$ m and standard deviation $e^{0.5}$ m.

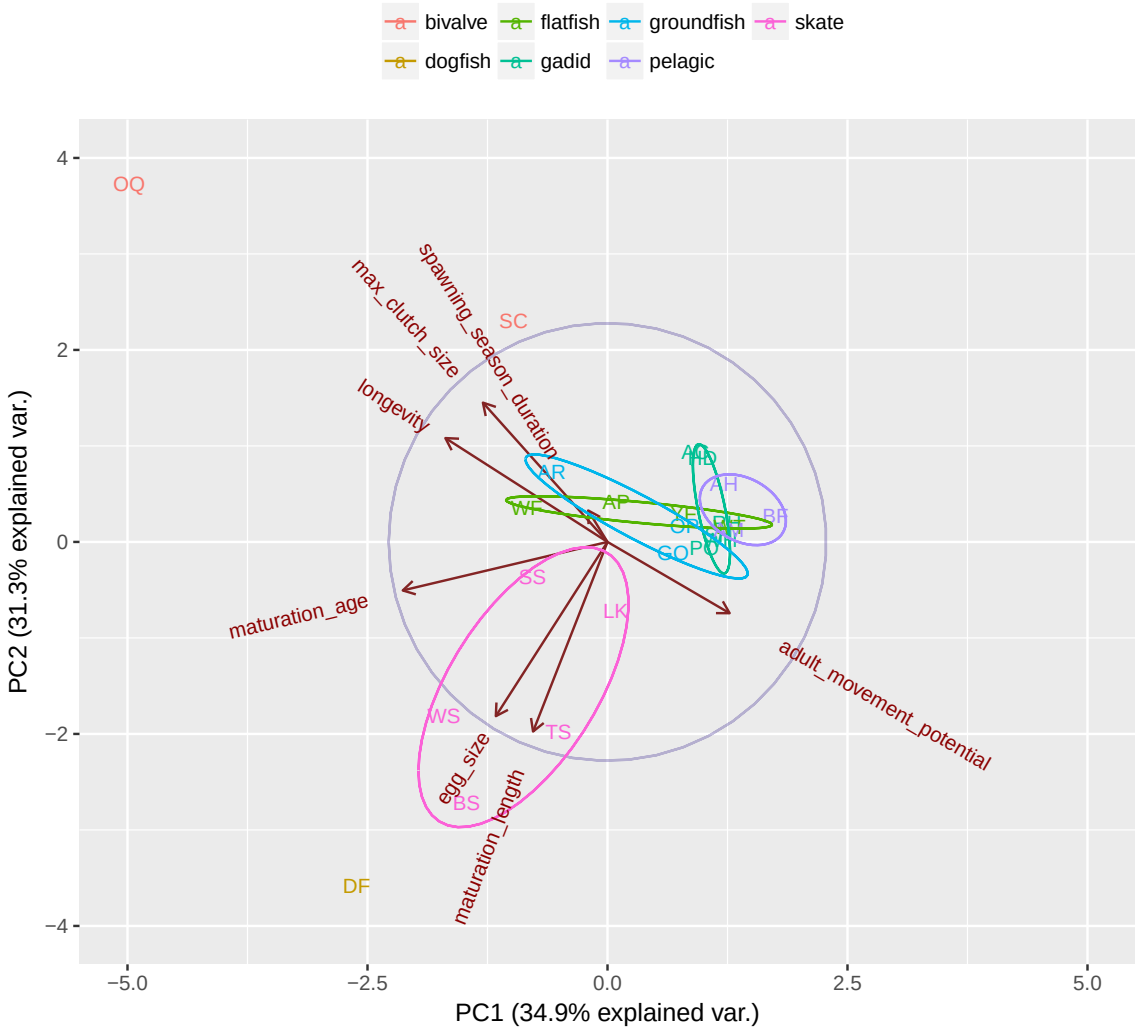

Figure E-1: The abbreviation for each species is plotted at its location along the first 2 principal components. The color of each abbreviation indicates the group to which is belongs, and the ovals encapsulate the species of that group. The axes corresponding to the original variables are plotted and labelled in brown.

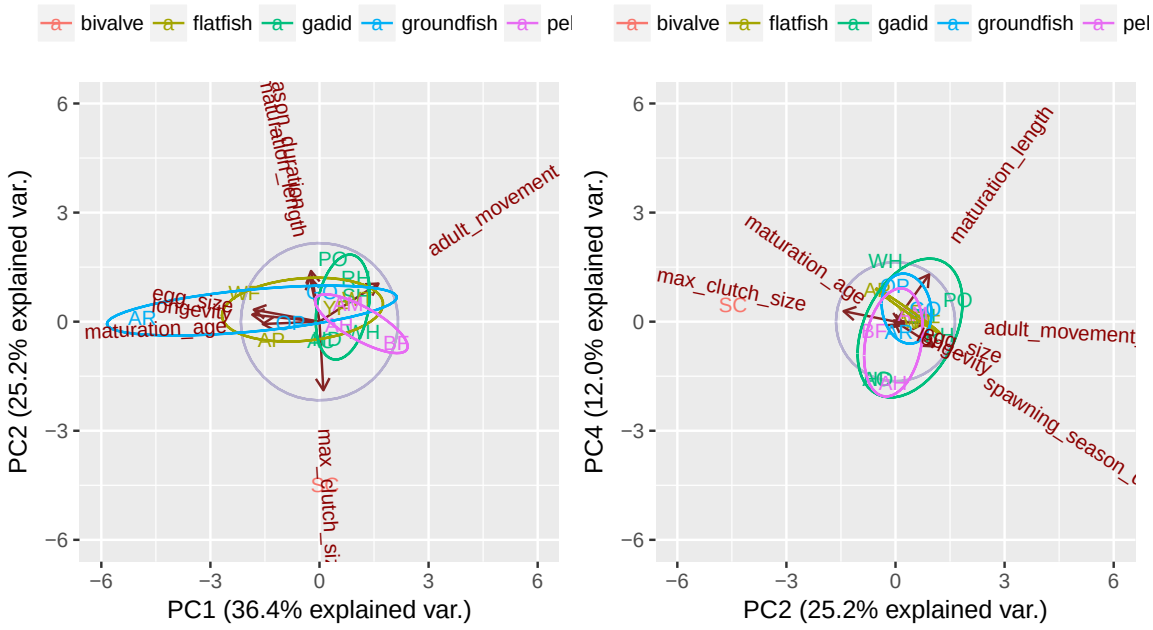

Figure E-2: Two possible pairings of the first 4 principal components are plotted. The location of each species is indicated by the appropriate abbreviation, and the color of the abbreviations and ovals indicates the group to which the species belongs. The axes corresponding to the original variables are plotted and labelled in brown. The cluster at (0, -1.5) on the right plot includes Atlantic herring, Atlantic cod, and haddock. Yellowtail flounder are located near (0, 0) in both plots.

E.4 Data

Table E.1: The taxonomic and grouping variables for each species are presented here. The groups were primarily taken from Liu et al. (2012)

Common name	Scientific Name	Family	Group
Atlantic cod	*Gadus morhua*	*Gadidae*	gadid
Haddock	*Melanogrammus aeglefinus*	*Gadidae*	gadid
Pollock	*Pollachius virens*	*Gadidae*	gadid
Silver hake	*Merluccius bilinearis*	*Gadidae*	gadid
Red hake	*Urophycis chuss*	*Gadidae*	gadid
White hake	*Urophycis tenuis*	*Gadidae*	gadid
Yellowtail flounder	*Limanda ferruginea*	*Paralichthyidae*	flatfish
Winter flounder	*Pseudopleuronectes americanus*	*Pleuronectidae*	flatfish
American plaice	*Hippoglossoides platessoides*	*Pleuronectidae*	flatfish
Witch flounder	*Glyptocephalus cynoglossus*	*Pleuronectidae*	flatfish
Ocean pout	*Macrozoarces americanus*	*Zoarcidae*	groundfish
Goosefish	*Lophius americanus*	*Lophiidae*	groundfish
Acadian redfish	*Sebastes fasciatus*	*Sebastidae*	groundfish
Little skate	*Leucoraja erinacea*	*Rajidae*	skate
Winter skate	*Leucoraja ocellata*	*Rajidae*	skate
Thorny skate	*Amblyraja radiata*	*Rajidae*	skate
Barndoor skate	*Dipturus laevis*	*Rajidae*	skate
Smooth skate	*Malacoraja senta*	*Rajidae*	skate
Atlantic mackerel	*Scomber scombrus*	*Scombridae*	pelagic
Butterfish	*Peprilus triacanthus*	*Stromateidae*	pelagic
Atlantic herring	*Clupea harengus*	*Clupeidae*	pelagic
Spiny dogfish	*Squalus acanthias*	*Squalidae*	dogfish
Ocean quahog	*Arctica islandica*	*Arcticidae*	bivalve
Sea scallop	*Placopecten magellanicus*	*Pectinidae*	bivalve

Table E.2: The first of two subsets of the variables for each species is reported here. AMP is the adult movement potential, A_{50} is the age at which the 50^{th} percentile of females mature, L_{50} is the length in mm at which the 50^{th} percentile of females mature, the maximum observed length is recorded in mm, and egg size is reported in μm. The maximum clutch sizes for AH and SS were used for BF and OQ respectively because values could not be located for the latter species.

Abbrv.	AMP	Long.	A_{50}	L_{50}	Max Length	Egg size	Max clutch size
AC	med	20	2	36		1450	9000000
HD	med	17	2	35	1000	1460	800000
PO	high	18	2	404	1200	1450	4000000
SH	high	14	2	275	780	457	391700
RH	high	14	1	269	630	800	400000
WH	high	20	1	350	1350	0	30000000
YF	med	12	2	290	600	900	200000
WT	high	14	1	256		795	3300000
AP	low	20	3	298	700	2290	1500000
WF	low	18	7	335	780	1075	900000
OP	med	20	2	313	980	3495	4200
GO	high	11	5	300	1400	1700	3200000
AR	low	50	5	223	450	6000	20000
LK	med	8	4	500		53500	30
WS	low	21	7	850	1500	125500	52
TS	high	20	7	840	1020	72000	20
BS	med	18	8	1020		128000	47
SS	low	14	5	560	577	55500	51
AM	high	17	1	257		1145	1980000
BF	high	3	1	120	305	750	200000
AH	high	18	3	26	390	1200	200000
DF	high	40	12	780	1250	265000	7
OQ	low	225	12	49	140	87	270000000
SC	low	12	4	80	170	66	270000000

Table E.3: The second of two subsets of the variables for each species is reported here. Depth is the depth or range of depths ([min, max]) in m at which larvae are generally observed, PLD is the range of pelagic larval durations in days, the spawning seasons give the days during which spawning is likely to take place, and the primary spawning season is indicated when more spawning occurs during one season than the other. Note that scallops generally seek the pycnocline, but a representative depth of 40 m was used here instead to facilitate quantitative analysis.

Abbrv.	Depth	PLD	Spring Spawning	Fall Spawning	Primary
AC	[30, 90]		[15-Nov, 15-May]		Spring
HD	[10, 50]	[30, 42]	[01-Jan, 30-Jun]		Spring
PO	[50, 90]	[90, 120]		[01-Sep, 30-Apr]	Fall
SH	[60, 130]	[34, 35]	[01-May, 31-Oct]		Spring
RH		60	[01-Apr, 30-Nov]		Spring
WH	[10, 150]	[10, 36]	[01-Apr, 31-May]	[01-Aug, 30-Sep]	Fall
YF	[10, 90]	[60, 61]	[01-Mar, 31-Aug]		Spring
WT		[35, 56]	[01-Nov, 30-Apr]		Spring
AP	[50, 90]		[01-Mar, 15-Jun]		Spring
WF	[10, 210]	[120, 365]	[01-Mar, 30-Nov]		Spring
OP				[01-Sep, 30-Nov]	Fall
GO	[30, 90]		[01-Apr, 30-Sep]		Spring
AR	[0, 30]	120	[01-Apr, 31-Aug]		Spring
LK			[01-Apr, 31-May]	[01-Oct, 31-Jan]	Both
WS				[30-Jun, 31-Jan]	Fall
TS		[720, 900]		[01-Aug, 30-Nov]	Fall
BS				[01-Dec, 31-Jan]	Fall
SS				[01-Jan, 31-Dec]	Both
AM	[10, 50]	[36, 60]	[01-Apr, 31-Aug]		Spring
BF	[10, 120]	[33, 55]	[01-Jun, 31-Aug]		Spring
AH	[50, 90]	[120, 240]		[01-Jul, 31-Dec]	Fall
DF				[01-Aug, 30-Nov]	Both
OQ	[1, 40]	[32, 60]	[01-May, 30-Nov]		Spring
SC	[0, 40]	[30, 45]	[01-May, 30-Jun]	[01-Aug, 15-Oct]	Fall

Table E.4: The sources for the data presented in Table E.2 and Table E.3 are listed here.

Abbrv	Sources
AC	(Food & Agriculture Organization of the United Nations, 2017a)
HD	(Blanchard et al., 2003)
PO	(Food & Agriculture Organization of the United Nations, 2017b)
SH	(Col and Traver, 2006; Lock and Packer, 2004; Cornell University Cooperative Extension, 2017; Steves and Cowen, 2000)
RH	(Steimle et al., 1999a)
WH	(Beacham and Nepszy, 1980)
YF	(Cadrin, 2010; Johnson et al., 1999; Zamarro, 1991)
WT	(Pereira et al., 1999)
AP	(Johnson, 2004)
WF	(Cargnelli et al., 1999a)
OP	(Steimle et al., 1999b)
GO	(Steimle et al., 1999c)
AR	(Pikanowski et al., 1999)
LK	(Packer et al., 2003b)
WS	(Centre for Marine Biodiversity, 2017)
TS	(Packer et al., 2003c)
BS	(Packer et al., 2003a)
SS	(Centre for Marine Biodiversity, 2017)
AM	(Studholme et al., 1999)
BF	(Cross et al., 1999)
AH	(Kelly and Stephenson, 1985)
DF	(Stehlik, 2007)
OQ	(Cargnelli et al., 1999b)
SC	(Tian et al., 2009a)

Appendix F

Species Parameters

This appendix presents the parameters for the species simulated in chapter 5.

Table F.1: The spawning parameters for each species are presented here. The spawning time is reported as the mean ± standard deviation of the normal distribution for the spawning time in days after midnight on 1 Jan 1995.

Species	Spawning Time	Spawning Substrate	Max Spawning Depth
1	157.80 ± 32.98	both	68.93 m
2	5.62 ± 27.21	sand	100.60 m
3	235.25 ± 6.30	sand	50.08 m
4	239.56 ± 40.57	sand	94.69 m
5	33.42 ± 98.87	sand	179.42 m
6	270.09 ± 50.39	both	102.40 m
7	357.89 ± 32.47	sand	59.80 m
8	179.16 ± 31.67	both	95.05 m
9	88.99 ± 75.97	both	110.68 m
10	37.68 ± 15.07	both	78.65 m
11	249.17 ± 15.87	sand	151.42 m
12	168.06 ± 21.61	sand	163.36 m
13	286.50 ± 58.23	sand	87.56 m
14	278.55 ± 14.95	sand	93.33 m
15	82.66 ± 8.11	sand	33.45 m
16	143.38 ± 12.32	gravel	26.28 m
17	309.78 ± 8.32	both	98.40 m
18	341.29 ± 7.52	gravel	99.32 m
19	92.07 ± 25.47	both	42.00 m
20	215.44 ± 29.86	gravel	33.99 m
21	175.27 ± 53.81	both	128.65 m
22	213.55 ± 19.84	gravel	112.28 m
23	290.61 ± 17.89	gravel	137.88 m
24	267.87 ± 15.43	gravel	80.74 m
25	317.82 ± 30.95	gravel	77.45 m

Continued on next page

Table F.1 – continued from previous page

Species	Spawning Time	Spawning Substrate	Max Spawning Depth
26	312.08 ± 18.03	gravel	155.64 m
27	4.13 ± 9.65	gravel	108.97 m
28	240.17 ± 19.31	gravel	151.36 m
29	343.99 ± 99.75	gravel	200.86 m
30	77.49 ± 37.34	both	67.11 m
31	215.37 ± 26.84	sand	74.24 m
32	312.77 ± 24.26	sand	62.48 m
33	102.25 ± 151.43	both	78.80 m
34	78.09 ± 9.09	sand	57.90 m
35	119.20 ± 24.40	gravel	108.90 m
36	102.95 ± 17.78	both	125.35 m
37	124.93 ± 40.46	gravel	115.20 m
38	89.24 ± 35.32	both	117.02 m
39	10.69 ± 10.06	gravel	161.17 m
40	92.85 ± 15.79	both	66.31 m
41	226.14 ± 28.53	sand	35.16 m
42	161.89 ± 53.14	gravel	107.57 m
43	17.65 ± 9.37	gravel	74.16 m
44	329.90 ± 8.33	gravel	63.42 m
45	196.08 ± 33.61	sand	144.75 m
46	349.43 ± 27.85	sand	182.49 m
47	281.27 ± 78.92	sand	91.21 m
48	111.40 ± 9.79	sand	150.28 m
49	110.13 ± 3.32	gravel	173.92 m
50	262.73 ± 12.87	sand	71.33 m
51	330.69 ± 51.04	both	70.08 m
52	186.67 ± 18.91	sand	56.28 m
53	186.96 ± 44.40	sand	71.80 m
54	155.76 ± 22.45	sand	146.42 m
55	143.72 ± 5.54	sand	105.19 m
56	300.20 ± 24.36	gravel	137.59 m
57	102.16 ± 25.41	both	51.58 m
58	121.74 ± 85.36	sand	131.81 m
59	269.05 ± 8.51	both	68.71 m
60	30.38 ± 13.93	both	90.67 m
61	131.57 ± 7.04	both	62.46 m
62	214.65 ± 19.96	gravel	80.00 m
63	256.15 ± 19.14	gravel	336.46 m
64	155.80 ± 20.70	both	120.96 m
65	326.52 ± 7.92	gravel	44.46 m
66	343.84 ± 5.97	sand	144.37 m
67	69.76 ± 11.57	both	142.23 m

Continued on next page

Table F.1 – continued from previous page

Species	Spawning Time	Spawning Substrate	Max Spawning Depth
68	270.35 ± 15.54	sand	154.46 m
69	116.38 ± 35.32	both	54.12 m
70	219.99 ± 42.21	gravel	95.75 m
71	212.82 ± 91.26	both	56.59 m
72	141.94 ± 4.60	gravel	60.16 m
73	197.17 ± 49.45	sand	72.10 m
74	135.60 ± 13.87	gravel	46.51 m
75	69.90 ± 35.41	gravel	82.77 m
76	116.10 ± 19.72	gravel	177.74 m
77	260.73 ± 83.93	sand	70.50 m
78	178.85 ± 17.94	sand	225.33 m
79	49.57 ± 20.89	sand	53.23 m
80	58.00 ± 7.87	gravel	176.82 m
81	207.46 ± 34.69	gravel	71.87 m
82	48.82 ± 8.22	sand	216.94 m
83	257.03 ± 37.66	gravel	127.11 m
84	21.55 ± 26.93	sand	370.73 m
85	309.98 ± 31.13	gravel	48.62 m
86	223.90 ± 36.92	both	82.18 m
87	328.97 ± 18.62	both	77.32 m
88	291.81 ± 30.00	both	66.45 m
89	235.16 ± 17.46	sand	95.24 m
90	25.42 ± 7.04	gravel	139.34 m
91	290.36 ± 23.10	gravel	119.07 m
92	328.51 ± 29.79	sand	58.27 m
93	204.06 ± 37.46	gravel	58.18 m
94	5.97 ± 23.55	sand	92.07 m
95	211.44 ± 23.76	gravel	59.24 m
96	97.94 ± 32.99	sand	170.26 m
97	20.70 ± 102.05	sand	124.94 m
98	39.80 ± 57.85	sand	21.93 m
99	304.03 ± 43.43	both	52.74 m
100	337.89 ± 29.03	sand	76.94 m
Yellowtail flounder	134.00 ± 21.00	sand	100.00 m
Sea scallop	134.00 ± 7.00	gravel	100.00 m
Haddock	73.00 ± 21.00	both	90.00 m
Atlantic herring	287.00 ± 21.00	gravel	80.00 m

Table F.2: The larval parameters for each species are presented here.

Species	Behavior	Minimum PLD	Competency Window
1	dvm	58.18 days	11.78 days
2	dvm	56.38 days	12.60 days
3	pycnocline	98.42 days	18.90 days
4	surface	49.08 days	9.61 days
5	passive	75.50 days	13.56 days
6	surface	52.47 days	13.43 days
7	dvm	53.72 days	6.39 days
8	dvm	65.26 days	16.72 days
9	passive	57.01 days	13.24 days
10	pycnocline	59.01 days	10.47 days
11	pycnocline	61.90 days	15.88 days
12	surface	50.85 days	7.13 days
13	dvm	47.18 days	6.73 days
14	pycnocline	63.03 days	14.20 days
15	pycnocline	62.24 days	9.61 days
16	pycnocline	57.78 days	12.79 days
17	surface	65.00 days	14.32 days
18	surface	78.54 days	19.74 days
19	pycnocline	55.07 days	9.18 days
20	dvm	56.18 days	8.52 days
21	dvm	80.25 days	14.96 days
22	passive	55.85 days	9.83 days
23	passive	47.40 days	8.97 days
24	pycnocline	70.04 days	13.19 days
25	passive	48.16 days	9.50 days
26	dvm	46.76 days	9.17 days
27	pycnocline	55.43 days	14.45 days
28	surface	46.00 days	10.43 days
29	dvm	61.30 days	14.76 days
30	surface	71.12 days	13.84 days
31	dvm	75.91 days	16.07 days
32	pycnocline	58.04 days	14.54 days
33	passive	64.78 days	13.99 days
34	passive	78.25 days	14.57 days
35	passive	81.76 days	16.10 days
36	surface	66.08 days	15.33 days
37	pycnocline	44.15 days	10.28 days
38	surface	44.35 days	9.38 days
39	passive	53.21 days	8.01 days
40	surface	57.64 days	12.17 days
41	dvm	62.67 days	14.69 days

Continued on next page

Table F.2 – continued from previous page

Species	Behavior	Minimum PLD	Competency Window
42	pycnocline	88.88 days	17.73 days
43	surface	53.33 days	7.51 days
44	surface	40.08 days	6.56 days
45	surface	59.29 days	11.41 days
46	pycnocline	50.44 days	9.42 days
47	dvm	70.22 days	17.79 days
48	passive	62.57 days	13.90 days
49	passive	70.44 days	11.33 days
50	passive	55.13 days	10.52 days
51	surface	95.82 days	18.94 days
52	pycnocline	48.38 days	13.66 days
53	surface	80.76 days	15.55 days
54	pycnocline	56.30 days	14.69 days
55	surface	66.03 days	14.62 days
56	dvm	77.48 days	15.43 days
57	pycnocline	52.44 days	8.69 days
58	pycnocline	55.15 days	10.40 days
59	surface	40.74 days	4.19 days
60	pycnocline	67.33 days	14.75 days
61	dvm	50.96 days	12.92 days
62	dvm	56.42 days	11.14 days
63	surface	49.51 days	5.23 days
64	dvm	56.38 days	11.86 days
65	dvm	55.07 days	10.38 days
66	pycnocline	72.82 days	14.39 days
67	passive	57.13 days	15.70 days
68	pycnocline	105.96 days	20.78 days
69	passive	67.43 days	10.27 days
70	passive	40.10 days	8.89 days
71	passive	64.27 days	13.02 days
72	pycnocline	51.26 days	11.78 days
73	pycnocline	30.58 days	5.03 days
74	surface	41.79 days	10.16 days
75	passive	77.06 days	12.88 days
76	passive	82.90 days	13.75 days
77	surface	58.64 days	11.11 days
78	passive	57.07 days	11.90 days
79	dvm	43.66 days	7.68 days
80	surface	64.35 days	14.21 days
81	surface	59.19 days	11.91 days
82	passive	56.08 days	11.78 days
83	dvm	47.45 days	10.65 days

Continued on next page

Table F.2 – continued from previous page

Species	Behavior	Minimum PLD	Competency Window
84	dvm	63.79 days	12.91 days
85	passive	50.40 days	8.72 days
86	passive	63.95 days	12.35 days
87	pycnocline	56.52 days	10.85 days
88	dvm	48.57 days	10.54 days
89	pycnocline	69.81 days	15.25 days
90	surface	51.03 days	13.90 days
91	pycnocline	67.40 days	12.97 days
92	pycnocline	65.79 days	11.51 days
93	dvm	64.14 days	13.55 days
94	passive	62.48 days	9.56 days
95	pycnocline	44.87 days	3.72 days
96	surface	47.89 days	11.43 days
97	dvm	45.18 days	8.53 days
98	surface	67.43 days	16.05 days
99	dvm	55.04 days	11.38 days
100	pycnocline	53.50 days	8.75 days
Yellowtail flounder	passive	55.00 days	20.00 days
Sea scallop	pycnocline	30.00 days	15.00 days
Haddock	passive	40.00 days	20.00 days
Atlantic herring	dvm	120.00 days	120.00 days

Table F.3: The settlement parameters for each species are presented here. The settlement probabilities are reported for fine sand, coarse sand, and gravel in that order and are normalized to sum to 1.

Species	Max settlement depth	Settlement probabilities
1	54.63 m	0.3642, 0.1721, 0.4637
2	80.46 m	0.1051, 0.5115, 0.3834
3	105.63 m	0.1637, 0.0730, 0.7633
4	97.64 m	0.4227, 0.0990, 0.4783
5	243.56 m	0.1057, 0.4253, 0.4690
6	115.58 m	0.2789, 0.5590, 0.1621
7	68.80 m	0.1853, 0.5628, 0.2519
8	68.77 m	0.4117, 0.4956, 0.0927
9	161.63 m	0.0966, 0.4336, 0.4698
10	78.44 m	0.4036, 0.3417, 0.2547
11	50.50 m	0.3739, 0.3915, 0.2345
12	61.25 m	0.1524, 0.4175, 0.4301
13	62.92 m	0.9694, 0.0142, 0.0164
14	111.07 m	0.4640, 0.3031, 0.2328

Continued on next page

Table F.3 – continued from previous page

Species	Max settlement depth	Settlement probabilities
15	116.01 m	0.1096, 0.4529, 0.4375
16	40.75 m	0.3748, 0.2340, 0.3911
17	148.79 m	0.5144, 0.2833, 0.2023
18	72.34 m	0.6504, 0.2448, 0.1047
19	54.86 m	0.3441, 0.4558, 0.2001
20	57.21 m	0.4215, 0.5224, 0.0561
21	118.21 m	0.0860, 0.4507, 0.4633
22	88.30 m	0.4575, 0.5317, 0.0109
23	102.39 m	0.5732, 0.1462, 0.2805
24	63.05 m	0.0581, 0.8562, 0.0857
25	50.64 m	0.2258, 0.4210, 0.3532
26	60.18 m	0.3687, 0.5774, 0.0539
27	94.88 m	0.2445, 0.1607, 0.5948
28	90.35 m	0.7483, 0.0055, 0.2462
29	84.66 m	0.1656, 0.5970, 0.2373
30	137.60 m	0.0765, 0.7281, 0.1954
31	77.38 m	0.1067, 0.3101, 0.5832
32	83.14 m	0.1855, 0.2955, 0.5191
33	101.82 m	0.0343, 0.9584, 0.0073
34	64.17 m	0.4122, 0.1000, 0.4878
35	99.34 m	0.3226, 0.5524, 0.1249
36	130.69 m	0.3367, 0.5633, 0.1000
37	64.22 m	0.1147, 0.5115, 0.3738
38	153.18 m	0.2439, 0.2524, 0.5038
39	117.51 m	0.0582, 0.4725, 0.4693
40	86.00 m	0.9409, 0.0221, 0.0370
41	54.31 m	0.7071, 0.0386, 0.2543
42	125.14 m	0.6333, 0.0433, 0.3234
43	87.74 m	0.2179, 0.7385, 0.0435
44	73.39 m	0.6910, 0.2988, 0.0102
45	81.76 m	0.1835, 0.0268, 0.7897
46	50.90 m	0.1282, 0.3188, 0.5530
47	93.92 m	0.1719, 0.5696, 0.2585
48	67.51 m	0.2679, 0.2550, 0.4771
49	90.08 m	0.1275, 0.5767, 0.2958
50	46.97 m	0.1807, 0.7031, 0.1162
51	100.56 m	0.3728, 0.1729, 0.4543
52	70.43 m	0.3603, 0.6109, 0.0288
53	108.62 m	0.3344, 0.2581, 0.4075
54	79.30 m	0.0657, 0.1660, 0.7682
55	52.34 m	0.1318, 0.5453, 0.3229
56	157.59 m	0.8805, 0.0676, 0.0519

Continued on next page

Table F.3 – continued from previous page

Species	Max settlement depth	Settlement probabilities
57	145.25 m	0.6623, 0.2149, 0.1227
58	123.60 m	0.1689, 0.1224, 0.7087
59	117.47 m	0.1147, 0.7643, 0.1210
60	49.44 m	0.3658, 0.2419, 0.3922
61	202.13 m	0.0420, 0.1336, 0.8244
62	197.62 m	0.3405, 0.2624, 0.3971
63	124.20 m	0.5298, 0.3208, 0.1494
64	87.24 m	0.8117, 0.0298, 0.1585
65	95.94 m	0.1849, 0.0612, 0.7539
66	87.90 m	0.6011, 0.1853, 0.2136
67	81.20 m	0.4094, 0.5613, 0.0293
68	171.80 m	0.4059, 0.1864, 0.4077
69	50.78 m	0.6722, 0.1631, 0.1648
70	84.20 m	0.4857, 0.0068, 0.5075
71	38.85 m	0.6315, 0.1759, 0.1925
72	380.51 m	0.7047, 0.0281, 0.2672
73	88.79 m	0.7676, 0.0761, 0.1564
74	111.18 m	0.5528, 0.2486, 0.1986
75	117.13 m	0.3019, 0.2797, 0.4184
76	167.61 m	0.5066, 0.1055, 0.3879
77	76.21 m	0.5236, 0.3164, 0.1600
78	95.63 m	0.1105, 0.1299, 0.7596
79	82.62 m	0.0378, 0.3758, 0.5864
80	151.55 m	0.2347, 0.7001, 0.0652
81	125.34 m	0.0695, 0.1892, 0.7413
82	183.18 m	0.5334, 0.2002, 0.2663
83	94.36 m	0.1023, 0.8966, 0.0012
84	105.74 m	0.6627, 0.2447, 0.0925
85	179.56 m	0.1690, 0.4239, 0.4071
86	171.45 m	0.0647, 0.3484, 0.5870
87	132.65 m	0.0391, 0.1737, 0.7872
88	163.34 m	0.3868, 0.3325, 0.2807
89	137.89 m	0.1989, 0.2219, 0.5792
90	159.80 m	0.0466, 0.8557, 0.0976
91	26.72 m	0.4530, 0.2951, 0.2519
92	56.72 m	0.0113, 0.4533, 0.5354
93	65.08 m	0.4402, 0.1323, 0.4275
94	160.14 m	0.0601, 0.7872, 0.1527
95	70.63 m	0.0678, 0.4997, 0.4325
96	47.82 m	0.8078, 0.1426, 0.0496
97	43.94 m	0.2255, 0.5318, 0.2427
98	60.97 m	0.2033, 0.3446, 0.4521

Continued on next page

Table F.3 – continued from previous page

Species	Max settlement depth	Settlement probabilities
99	89.10 m	0.1052, 0.1636, 0.7312
100	99.17 m	0.1511, 0.6101, 0.2388
Yellowtail flounder	100.00 m	0.4762, 0.4762, 0.0476
Sea scallop	100.00 m	0.0385, 0.1923, 0.7692
Haddock	90.00 m	0.3333, 0.3333, 0.3333
Atlantic herring	500.00 m	0.3333, 0.3333, 0.3333

Bibliography

Almany, G. R., Berumen, M. L., Thorrold, S. R., Planes, S., and Jones, G. P. (2007). Local replenishment of coral reef fish populations in a marine reserve. *Science*, 316(5825):742–744.

Baranyi, G., Saura, S., Podani, J., and Jordán, F. (2011). Contribution of habitat patches to network connectivity: Redundancy and uniqueness of topological indices. *Ecological Indicators*, 11(5):1301 – 1310.

Barton, A. D., Pershing, A. J., Litchman, E., Record, N. R., Edwards, K. F., Finkel, Z. V., Kiørboe, T., and Ward, B. A. (2013). The biogeography of marine plankton traits. *Ecology Letters*, 16(4):522–534.

Beacham, T. D. and Nepszy, S. J. (1980). Some aspects of the biology of white hake, *Urophycis tenuis*, in the Southern Gulf of St. Lawrence. *Journal of the Northwest Fishery Science*, 1:49–54.

Beazley, D. M. (1996). SWIG: An easy to use tool for integrating scripting languages with C and C++. In *Proceedings of the 4th Conference on USENIX Tcl/Tk Workshop, 1996 - Volume 4*, TCLTK'96, pages 15–15, Berkeley, CA, USA. USENIX Association.

Blanchard, J. L., Frank, K. T., and Simon, J. E. (2003). Effects of condition on fecundity and total egg production of eastern Scotian Shelf haddock (*Melanogrammus aeglefinus*). *Canadian Journal of Fisheries and Aquatic Sciences*, 60:321–332.

Bleck, R., Halliwell, G., Wallcraft, A., Carroll, S., Kelly, K., and Rushing, K. (2002). *HYbrid Coordinate Ocean Model (HYCOM) User's Manual*, 2.0.01 edition.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

Boucher, J. M., Chen, C., Sun, Y., and Beardsley, R. C. (2013). Effects of interannual environmental variability on the transport-retention dynamics in haddock *Melanogrammus aeglefinus* larvae on Georges Bank. *Marine Ecology Progress Series*, 487:201–215.

Brickman, D., Adlandsvik, B., Thygesen, U. H., Parada, C., Rose, K., Hermann, A. J., and Edwards, K. (2009). Particle tracking. In North, E. W., Gallego, A., and Petitgas, P., editors, *Manual of recommended practices for modelling physical - biological interactions during fish early life*, chapter 2, pages 9–19. International Council for Exploration of the Sea.

Brickman, D. and Smith, P. C. (2002). Lagrangian stochastic modeling in coastal oceanography. *Journal of Atmospheric and Oceanic Technology*, 19(1):83–99.

Cadrin, S. X. (2010). Interdisciplinary analysis of yellowtail flounder stock structure off New England. *Reviews in Fisheries Science*, 18(3):281–299.

Cargnelli, L. M., Griesbach, S. J., Berrien, P. L., Morse, W. M., and Johnson, D. L. (1999a). Haddock, *Melanogrammus aeglefinus*, life history and habitat characteristics. Technical Report NMFS-NE-128, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Cargnelli, L. M., Griesbach, S. J., Packer, D. B., and Weissberger, E. (1999b). Ocean quahog, *Arctica islandica*, life history and habitat characteristics. Technical Report NMFS-NE-148, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Centre for Marine Biodiversity (2017). Reproduction. `http://www.marinebiodiversity.ca/skatesandrays/skate%20research/reproduction%20research.htm`. Accessed 31 May 2017.

Chen, C., Beardsley, R. C., and Cowles, G. (2006). An unstructured grid, finite-volume coastal ocean model (FVCOM) system. *Oceanography*, 19(1):78–89.

Chen, C., Beardsley, R. C., Hu, S., Xu, Q., and Lin, H. (2005). Using MM5 to hindcast the ocean surface forcing fields over the Gulf of Maine and Georges Bank region. *Journal of Atmospheric and Oceanic Technology*, 22(2):131–145.

Chen, C., Liu, H., and Beardsley, R. C. (2003). An unstructured grid, finite-volume, three-dimensional, primitive equations ocean model: Application to coastal ocean and estuaries. *Journal of Atmospheric and Oceanic Technology*, 20(1):159–186.

Churchill, J. H., Runge, J., and Chen, C. (2011). Processes controlling retention of spring-spawned Atlantic cod (*Gadus morhua*) in the western gulf of maine and their relationship to an index of recruitment success. *Fisheries Oceanography*, 20(1):32–46.

Col, L. and Traver, M. (2006). Status of fishery resources off the northeastern US : Silver hake (*Merluccius bilinearis*). `https://www.nefsc.noaa.gov/sos/spsyn/pg/silverhake/`. Revised December 2006.

Cornell University Cooperative Extension (2017). Silverhake / *Merluccius bilinearis*. https://s3.amazonaws.com/assets.cce.cornell.edu/attachments/3631/silverhake.pdf?1414173174. Accessed 31 May 2017.

Couturier, R. (2014). Presentation of the GPU architecture and of the CUDA environment. In Couturier, R., editor, *Designing Scientific Applications on GPUs*, chapter 1, pages 3–12. CRC Press, Boca Raton.

Cowen, R. K., Gawarkiewicz, G., Pineda, J., Thorrold, S. R., and Werner, F. E. (2007). Population connectivity in marine systems: An overview. *Oceanography*, 20(3):14–21.

Cowen, R. K. and Guigand, C. M. (2008). In situ ichthyoplankton imaging system (ISIIS): system design and preliminary results. *Limnology and Oceanography: Methods*, 6(2):126–132.

Cowen, R. K., Lwiza, K. M. M., Sponaugle, S., Paris, C. B., and Olson, D. B. (2000). Connectivity of marine populations: Open or closed? *Science*, 287(5454):857–859.

Cowen, R. K. and Sponaugle, S. (2009). Larval dispersal and marine population connectivity. *Annual Review of Marine Science*, 1(1):443–466.

Cowles, G. W. (2008). Parallelization of the FVCOM coastal ocean model. *The International Journal of High Performance Computing Applications*, 22(2):177–193.

Cowles, G. W., Lentz, S. J., Chen, C., Xu, Q., and Beardsley, R. C. (2008). Comparison of observed and model-computed low frequency circulation and hydrography on the New England Shelf. *Journal of Geophysical Research: Oceans*, 113(C9):n/a–n/a. C09015.

Cross, J. N., Zetlin, C. A., Berrien, P. L., Johnson, D. L., , and McBride, C. (1999). Butterfish, *Peprilus triacanthus*, life history and habitat characteristics. Technical Report NMFS-NE-145, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Dixson, D. L., Jones, G. P., Munday, P. L., Planes, S., Pratchett, M. S., Srinivasan, M., Syms, C., and Thorrold, S. R. (2008). Coral reef fish smell leaves to find island homes. *Proceedings of the Royal Society of London B: Biological Sciences*, 275(1653):2831–2839.

Dixson, D. L., Jones, G. P., Munday, P. L., Pratchett, M. S., Srinivasan, M., Planes, S., and Thorrold, S. R. (2011). Terrestrial chemical cues help coral reef fish larvae locate settlement habitat surrounding islands. *Ecology and Evolution*, 1(4):586–595.

Edler, D., Guedes, T., Zizka, A., Rosvall, M., and Antonelli, A. (2015). Infomap bioregions: Interactive mapping of biogeographical regions from species distributions. *ArXiv e-prints*.

Engsig-Karup, A. P., Glimberg, S. L., Nielsen, A. S., and Lindberg, O. (2014). Fast hydrodynamics on heterogeneous many-core hardware. In Couturier, R., editor, *Designing Scientific Applications on GPUs*, chapter 11, pages 251–294. CRC Press, Boca Raton.

Fogarty, M. J. and Botsford, L. W. (2007). Population connectivity and spatial management of marine fisheries. *Oceanography*, 20(3):112–123.

Follows, M. J., Dutkiewicz, S., Grant, S., and Chisholm, S. W. (2007). Emergent biogeography of microbial communities in a model ocean. *Science*, 315(5820):1843–1846.

Food & Agriculture Organization of the United Nations (2017a). *Gadus morhua*. `http://www.fao.org/fishery/culturedspecies/Gadus_morhua/en`. Accessed 31 May 2017.

Food & Agriculture Organization of the United Nations (2017b). *Melanogrammus aeglefinus*. `http://www.fao.org/fishery/species/2228/en`. Accessed 31 May 2017.

Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174.

Gallego, A. and North, E. W. (2009). Biological processes. In North, E. W., Gallego, A., and Petitgas, P., editors, *Manual of recommended practices for modelling physical - biological interactions during fish early life*, chapter 3, pages 20–59. International Council for Exploration of the Seas.

Gilbert, C., Gentleman, W., Johnson, C., DiBacco, C., Pringle, J., and Chen, C. (2010). Modelling dispersal of sea scallop (*Placopecten magellanicus*) larvae on Georges Bank: The influence of depth-distribution, planktonic duration and spawning seasonality. *Progress in Oceanography*, 87(1-4):37 – 48. 3rd {GLOBEC} OSM: From ecosystem function to ecosystem prediction.

Grimm, V. and Railsback, S. F. (2005). *Individual-based Modeling and Ecology*. Princeton University Press, Princeton.

Haller, G. (2015). Lagrangian coherent structures. *Annual Review of Fluid Mechanics*, 47(1):137–162.

Harrison, C. S. and Glatzmaier, G. A. (2012). Lagrangian coherent structures in the california current system – sensitivities and limitations. *Geophysical & Astrophysical Fluid Dynamics*, 106(1):22–44.

Harrison, C. S., Siegel, D. A., and Mitarai, S. (2013). Filamentation and eddy-eddy interactions in marine larval accumulation and transport. *Marine Ecology Progress Series*, 472:27–44.

Hart, D. R. and Chute, A. S. (2004). Sea scallop, *Placopecten magellanicus*, life history and habitat characteristics. Technical Report NMFS-NE-189, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Hastings, A. (1993). Complex interactions between dispersal and dynamics: Lessons from coupled logistic equations. *Ecology*, 74(5):pp. 1362–1372.

Herault, A., Bilotta, G., and Dalrymple, R. A. (2010). SPH on GPU with CUDA. *Journal of Hydraulic Research*, 48(sup1):74–79.

Hufnagl, M., Payne, M., Lacroix, G., Loes, J. B., Daewel, U., Dickey-Collas, M., Gerkema, T., Huret, M., Janssen, F., Kreus, M., Patsch, J., Pohlmann, T., Ruardij, P., Schrum, C., Skogen, M., Meinard, C. T., Petitgas, P., van Beek Jan, K., van der Veer Henk, W., and Callies, U. (2017). Variation that can be expected when using particle tracking models in connectivity studies. *Journal of Sea Research*.

Huret, M., Runge, J. A., Chen, C., Cowles, G., Xu, Q., and Pringle, J. M. (2007). Dispersal modeling of fish early life stages: sensitivity with application to Atlantic cod in the western Gulf of Maine. *Marine Ecology Progress Series*, 347:261–274.

Intel Corportation (2017). *Intel Xeon Processor E5-2650*. Intel Corporation.

Irisson, J.-O., Leis, J. M., Paris, C. B., and Browman, H. I. (2009). Behaviour and settlement. In North, E. W., Gallego, A., and Petitgas, P., editors, *Manual of recommended practices for modelling physical - biological interactions during fish early life*, chapter 3, pages 20–59. International Council for Exploration of the Seas.

Jacobi, M. N., Andrè, C., Döös, K., and Jonsson, P. R. (2012). Identification of subpopulations from connectivity matrices. *Ecography*, 35(11):1004–1016.

Ji, R., Ashjian, C. J., Campbell, R. G., Chen, C., Gao, G., Davis, C. S., Cowles, G. W., and Beardsley, R. C. (2012). Life history and biogeography of *Calanus* copepods in the arctic ocean: An individual-based modeling study. *Progress in Oceanography*, 96(1):40 – 56.

Johnson, D. L. (2004). American plaice, *Hippoglossoides platessoides*, life history and habitat characteristics. Technical Report NMFS-NE-187, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Johnson, D. L., Morse, W. W., Berrien, P. L., and Vitaliano, J. J. (1999). Yellowtail flounder, *Limanda ferruginea*, life history and habitat characteristics. Technical Report NMFS-NE-140, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Jones, B. T., Gyory, J., Grey, E. K., Bartlein, M., Ko, D. S., Nero, R. W., and Taylor, C. M. (2015). Transport of blue crab larvae in the northern Gulf of Mexico during the Deepwater Horizon oil spill. *Marine Ecology Progress Series*, 527:143–156.

Jones, B. T., Solow, A., and Ji, R. (2016). Resource allocation for lagrangian tracking. *Journal of Atmospheric and Oceanic Technology*, 33(6):1225–1235.

Kelly, K. H. and Stephenson, D. K. (1985). Fecundity of Atlantic herring (*Clupea harengus*) from three spawning areas in the western Gulf of Maine, 1969 and 1982. *Journal of the Northwest Fishery Science*, 6:149–155.

Kenchington, E. L., Patwary, M. U., Zouros, E., and Bird, C. J. (2006). Genetic differentiation in relation to marine landscape in a broadcast-spawning bivalve mollusc (*Placopecten magellanicus*). *Molecular Ecology*, 15(7):1781–1796.

Kininmonth, S. J., De'ath, G., and Possingham, H. P. (2010). Graph theoretic topology of the Great but small Barrier Reef world. *Theoretical Ecology*, 3(2):75–88.

Kleisner, K. M., Fogarty, M. J., McGee, S., Hare, J. A., Moret, S., Perretti, C. T., and Saba, V. S. (2017). Marine species distribution shifts on the U.S. Northeast continental shelf under continued ocean warming. *Progress in Oceanography*, 153:24 – 36.

Le Henaff, M., Kourafalou, V. H., Paris, C. B., Helgers, J., Aman, Z. M., Hogan, P. J., and Srinivasan, A. (2012). Surface evolution of the Deepwater Horizon oil spill patch: Combined effects of circulation and wind-induced drift. *Environmental Science & Technology*, 46(13):7267–7273. PMID: 22676453.

Lee, V. W., Kim, C., Chhugani, J., Deisher, M., Kim, D., Nguyen, A. D., Satish, N., Smelyanskiy, M., Chennupaty, S., Hammarlund, P., Singhal, R., and Dubey, P. (2010). Debunking the 100x GPU vs. CPU myth: an evaluation of throughput computing on CPU and GPU. *Proceedings of the 37th annual international symposium on Computer architecture*, pages 451–460.

Li, Y., Ji, R., Fratantoni, P. S., Chen, C., Hare, J. A., Davis, C. S., and Beardsley, R. C. (2014). Wind-induced interannual variability of sea level slope, along-shelf flow, and surface salinity on the Northwest Atlantic shelf. *Journal of Geophysical Research: Oceans*, 119(4):2462–2479.

Liu, H., Fogarty, M. J., Glaser, S. M., Altman, I., Hsieh, C.-h., Kaufman, L., Rosenberg, A. A., and Sugihara, G. (2012). Nonlinear dynamic features and copredictability of the Georges Bank fish community. *Marine Ecology Progress Series*, 464:195–207.

Lock, M. C. and Packer, D. B. (2004). Silver hake, *Merluccius bilinearis*, life history and habitat characteristics. Technical Report NMFS-NE-186, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Lowe, W. H. and Allendorf, F. W. (2010). What can genetics tell us about population connectivity? *Molecular Ecology*, 19(15):3038–3051.

Lynch, D. R., Greenberg, D. A., Bilgili, A., McGillicuddy, D. J. J., Manning, J. P., and Aretxabaleta, A. L. (2015). *Particles in the Coastal Ocean: Theory and Applications*. Cambridge University Press, Cambridge.

Maps, F., Pershing, A. J., and Record, N. R. (2012). A generalized approach for simulating growth and development in diverse marine copepod species. *ICES Journal of Marine Science*, 69(3):370.

Miller, C. B. and Wheeler, P. A. (2012). *Biological Oceanography*. Wiley-Blackwell, Hoboken, NJ, 2nd edition.

Mitarai, S., Siegel, D. A., Watson, J. R., Dong, C., and McWilliams, J. C. (2009). Quantifying connectivity in the coastal ocean with application to the Southern California Bight. *Journal of Geophysical Research: Oceans*, 114(C10):1–21.

Molkenthin, N., Kutza, H., Tupikina, L., Marwan, N., Donges, J. F., Feudel, U., Kurths, J., and Donner, R. V. (2016). Edge anisotropy and the geometric perspective on flow networks. *ArXiv e-prints*.

Moore, G. E. (1998). Cramming more components onto integrated circuits. *Proceedings of the IEEE*, 86(1):82–85.

North, E. W., Schlag, Z., Adams, E. E., Sherwood, C. R., He, R., Hyun, H., and Socolofsky, S. A. (2011). Simulating oil droplet dispersal from the Deepwater Horizon spill with a Lagrangian approach. In *Monitoring and Modeling the Deepwater Horizon Oil Spill: A Record-Breaking Enterprise*, volume 195, pages 217–226. Wiley.

NVIDIA Corportation (2012). *Tesla K20 GPU Accelerator*. NVIDIA, BD-06455-001_v05 edition.

Nye, J. A., Link, J. S., Hare, J. A., and Overholtz, W. J. (2009). Changing spatial distribution of fish stocks in relation to climate and population size on the Northeast United States continental shelf. *Marine Ecology Progress Series*, 393:111–129.

Owen, E. F. and Rawson, P. D. (2013). Small-scale spatial and temporal genetic structure of the Atlantic sea scallop (*Placopecten magellanicus*) in the inshore Gulf of Maine revealed using AFLPs. *Marine Biology*, 160(11):3015–3025.

Owens, J. D., Houston, M., Luebke, D., Green, S., Stone, J. E., and Phillips, J. C. (2008). GPU computing. *Proceedings of the IEEE*, 96(5):879–899.

Packer, D. B., Zetlin, C. A., and Vitaliano, J. J. (2003a). Barndoor skate, *Dipturus laevis*, life history and habitat characteristics. Technical Report NMFS-NE-173, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Packer, D. B., Zetlin, C. A., and Vitaliano, J. J. (2003b). Little skate, *Leucoraja erinacea*, life history and habitat characteristics. Technical Report NMFS-NE-175, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Packer, D. B., Zetlin, C. A., and Vitaliano, J. J. (2003c). Thorny skate, *Amblyraja radiata*, life history and habitat characteristics. Technical Report NMFS-NE-178, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Paris, C. B., Chérubin, L. M., and Cowen, R. K. (2007). Surfing, spinning, or diving from reef to reef: effects on population connectivity. *Marine Ecology Progress Series*, 347:285–300.

Paris, C. B., Helgers, J., van Sebille, E., and Srinivasan, A. (2013). Connectivity Modeling System: A probabilistic modeling tool for the multi-scale tracking of biotic and abiotic variability in the ocean. *Environmental Modelling & Software*, 42:47–54.

Pearce, C., Manuel, J., Gallager, S., Manning, D., O'Dor, R., and Bourget, E. (2004). Depth and timing of settlement of veligers from different populations of giant scallop, *Placopecten magellanicus (Gmelin)*, in thermally stratified mesocosms. *Journal of Experimental Marine Biology and Ecology*, 312(1):187 – 214.

Pereira, J. J., Goldberg, R., Ziskowski, J. J., Berrien, P. L., Morse, W. W., , and Johnson, D. L. (1999). Winter flounder, *Pseudopleuronectes americanus*, life history and habitat characteristics. Technical Report NMFS-NE-138, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Perry, A. L., Low, P. J., Ellis, J. R., and Reynolds, J. D. (2005). Climate change and distribution shifts in marine fishes. *Science*, 308(5730):1912–1915.

Petrik, C. M., Ji, R., and Davis, C. S. (2014). Interannual differences in larval haddock survival: hypothesis testing with a 3d biophysical model of Georges Bank. *Fisheries Oceanography*, 23(6):521–553.

Pikanowski, R. A., Morse, W. W., Berrien, P. L., Johnson, D. L., and McMillan, D. G. (1999). Redfish, *Sebastes* spp., life history and habitat characteristics. Technical Report NMFS-NE-132, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Pineda, J., Hare, J. A., and Sponaugle, S. (2007). Larval transport and dispersal in the coastal ocean and consequences for population connectivity. *Oceanography*, 20(3):22–39.

Planes, S., Jones, G. P., and Thorrold, S. R. (2009). Larval dispersal connects fish populations in a network of marine protected areas. *Proceedings of the National Academy of Sciences*.

Poppe, L., Williams, S., and Paskevich, V. (2005). CONMAPSG: Continental Margin Mapping (CONMAP) sediments grainsize distribution for the United States East Coast Continental Margin.

Posgay, J. and Norman, K. (1958). An observation on the spawning of the sea scallop, *Placopecten magellanicus (Gmelin)*, on Georges Bank. *Limnology and Oceanography*, 3:478.

R Core Team (2016). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rauber, T. and Runger, G. (2012). *Parallel Programming for Multicore and Cluster Systems*. Springer, Heidelberg, 2 edition.

Reid, R. N., Cargnelli, L. M., Griesbach, S. J., Packer, D. B., Johnson, D. L., Zetlin, C. A., Morse, W. M., and Berrien, P. L. (1999). Atlantic herring, *Clupea haraengus*, life history and habitat characteristics. Technical Report NMFS-NE-126, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Roache, P. J. (1998). Verification of codes and calculations. *AIAA Journal*, 36(5):696–702.

Rossi, V., Ser-Giacomi, E., López, C., and Hernández-García, E. (2014). Hydrodynamic provinces and oceanic connectivity from a transport network help designing marine reserves. *Geophysical Research Letters*, 41(8):2883–2891.

Rosvall, M. and Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123.

Samelson, R. and Wiggins, S. (2006). *Lagrangian Transport in Geophysical Jets and Waves*. Springer, New York.

Sanders, J. and Kandrot, E. (2010). *CUDA by Example*. Addison-Wesley, Upper Saddle River, NJ.

Schaub, M. T., Delvenne, J.-C., Yaliraki, S. N., and Barahona, M. (2012). Markov dynamics as a zooming lens for multiscale community detection: Non clique-like communities and the field-of-view limit. *PLoS ONE*, 7(2):1–11.

Schlag, Z. R. and North, E. W. (2012). Lagrangian TRANSport model (LTRANS v.2) User's Guide.

Ser-Giacomi, E., Rossi, V., López, C., and Hernández-García, E. (2015). Flow networks: A characterization of geophysical fluid transport. *Chaos*, 25(3):036404.

Shadden, S. C., Lekien, F., and Marsden, J. E. (2005). Definition and properties of lagrangian coherent structures from finite-time lyapunov exponents in two-dimensional aperiodic flows. *Physica D: Nonlinear Phenomena*, 212(3–4):271 – 304.

Shanks, A. L. (2009). Pelagic larval duration and dispersal distance revisited. *The Biological Bulletin*, 216(3):373–385.

Shanks, A. L., Grantham, B. A., and Carr, M. H. (2003). Propagule dispersal distance and the size and spacing of marine reserves. *Ecological Applications*, 13(sp1):159–169.

Shchepetkin, A. F. and McWilliams, J. C. (2005). The regional oceanic modeling system (ROMS): a split-explicit, free-surface, topography-following-coordinate oceanic model. *Ocean Modelling*, 9(4):347–404.

Simons, R. D., Siegel, D. A., and Brown, K. S. (2013). Model sensitivity and robustness in the estimation of larval transport: A study of particle tracking parameters. *Journal of Marine Systems*, 119–120(0):19 – 29.

Simpson, M. R. and Walsh, S. J. (2004). Changes in the spatial structure of grand bank yellowtail flounder: testing MacCall's basin hypothesis. *Journal of Sea Research*, 51(3–4):199 – 210. Proceedings of the Fifth International Symposium on Flatfish Ecology, Part {II}.

Slatkin, M. (1987). Gene flow and the geographic structure of natural populations. *Science*, 236(4803):787–792.

Staaterman, E. and Paris, C. B. (2014). Modelling larval fish navigation: the way forward. *ICES Journal of Marine Science*, 71(4):918.

Staaterman, E., Paris, C. B., and Helgers, J. (2012). Orientation behavior in fish larvae: A missing piece to Hjort's critical period hypothesis. *Journal of Theoretical Biology*, 304:188–196.

Stehlik, L. L. (2007). Spiny dogfish, *Squalus acanthias*, life history and habitat characteristics. Technical Report NMFS-NE-203, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Steimle, F. W., Morse, W. W., Berrien, P. L., and Johnson, D. L. (1999a). Red hake, *Urophycis chuss*, life history and habitat characteristics. Technical Report NMFS-NE-133, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Steimle, F. W., Morse, W. W., Berrien, P. L., Johnson, D. L., and Zetlin, C. A. (1999b). Ocean pout, *Macrozoarces americanus*, life history and habitat characteristics. Technical Report NMFS-NE-129, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Steimle, F. W., Morse, W. W., and Johnson, D. L. (1999c). Goosefish, *Lophius americanus*, life history and habitat characteristics. Technical Report NMFS-NE-127, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Stephenson, R. and Power, M. (1988). Semidiel vertical movements in Atlantic herring *Clupea harengus* larvae: a mechanism for larval retention? *Marine Ecology Progress Series*, 50:3–11.

Steves, B. P. and Cowen, R. P. (2000). Settlement, growth, and movement of silver hake *Merluccius bilinearis* in nursery habitat on the New York Bight continental shelf. *Marine Ecology Progress Series*, 196:279–290.

Studholme, A. L., Packer, D. B., Berrien, P. L., Johnson, D. L., Zetlin, C. A., and Morse, W. W. (1999). Atlantic mackerel, *Scomber scombrus*, life history and habitat characteristics. Technical Report NMFS-NE-141, U. S. Department of Commerce–National Oceanic and Atmospheric Administration–National Marine Fisheries Service–Northeast Fisheries Science Center, Woods Hole, Massachusetts.

Sun, Y., Chen, C., Beardsley, R. C., Ullman, D., Butman, B., and Lin, H. (2016). Surface circulation in Block Island Sound and adjacent coastal and shelf regions: A FVCOM-CODAR comparison. *Progress in Oceanography*, 143:26 – 45.

Sun, Y., Chen, C., Beardsley, R. C., Xu, Q., Qi, J., and Lin, H. (2013). Impact of current-wave interaction on storm surge simulation: A case study for Hurricane Bob. *Journal of Geophysical Research: Oceans*, 118(5):2685–2701.

Tanenbaum, A. S. and Bos, H. (2015). *Modern Operating Systems*. Pearson, Boston, 4 edition.

Thomas, C. J., Lambrechts, J., Wolanski, E., Traag, V. A., Blondel, V. D., Deleersnijder, E., and Hanert, E. (2014). Numerical modelling and graph theory tools to study ecological connectivity in the Great Barrier Reef. *Ecological Modelling*, 272:160 – 174.

Thorrold, S. R., Zacherl, D. C., and Levin, L. A. (2007). Population connectivity and larval dispersal using geochemical signatures in calcified structures. *Oceanography*, 20.

Tian, R. C., Chen, C., Stokesbury, K. D. E., Rothschild, B. J., Cowles, G. W., Xu, Q., Hu, S., Harris, B. P., and Marino, M. C. (2009a). Dispersal and settlement

of sea scallop larvae spawned in the fishery closed areas on Georges Bank. *ICES Journal of Marine Science: Journal du Conseil*.

Tian, R. C., Chen, C., Stokesbury, K. D. E., Rothschild, B. J., Cowles, G. W., Xu, Q., Hu, S., Harris, B. P., and Marino, M. C. I. (2009b). Modeling the connectivity between sea scallop populations in the Middle Atlantic Bight and over Georges Bank. *Marine Ecology Progress Series*, 380:147–160.

Tian, R. C., Chen, C., Stokesbury, K. D. E., Rothschild, B. J., Xu, Q., Hu, S., Cowles, G., Harris, B. P., and Marino II, M. C. (2009c). Sensitivity analysis of sea scallop (*Placopecten magellanicus*) larvae trajectories to hydrodynamic model configuration on Georges Bank and adjacent coastal regions. *Fisheries Oceanography*, 18(3):173–184.

Townsend, D. W., Thomas, A. C., Mayer, L. M., Thomas, M. A., and Quinlan, J. A. (2004). Oceanography of the Northwest Atlantic continental shelf. In Robinson, A. and Brink, K., editors, *The Sea: The Global Coastal Ocean: Interdisciplinary Regional Studies and Syntheses*, chapter 5, pages 1–57. Harvard University Press.

Traag, V. A., Van Dooren, P., and Nesterov, Y. (2011). Narrow scope for resolution-limit-free community detection. *Phys. Rev. E*, 84:016114.

Treml, E. A., Ford, J. R., Black, K. P., and Swearer, S. E. (2015). Identifying the key biophysical drivers, connectivity outcomes, and metapopulation consequences of larval dispersal in the sea. *Movement Ecology*, 3(1):17.

Treml, E. A., Halpin, P. N., Urban, D. L., and Pratson, L. F. (2008). Modeling population connectivity by ocean currents, a graph-theoretic approach for marine conservation. *Landscape Ecology*, 23(1):19–36.

Treml, E. A., Roberts, J. J., Chao, Y., Halpin, P. N., Possingham, H. P., and Riginos, C. (2012). Reproductive output and duration of the pelagic larval stage determine seascape-wide connectivity of marine populations. *Integrative and Comparative Biology*, 52(4):525–537.

Van Wyngaarden, M., Snelgrove, P. V. R., DiBacco, C., Hamilton, L. C., Rodríguez-Ezpeleta, N., Jeffery, N. W., Stanley, R. R. E., and Bradbury, I. R. (2017). Identifying patterns of dispersal, connectivity and selection in the sea scallop, *Placopecten magellanicus*, using RADseq-derived SNPs. *Evolutionary Applications*, 10(1):102–117.

Versteeg, H. K. and Malalasekera, W. (2007). *An Introduction to Computational Fluid Dynamics*. Pearson Education, Harlow, England, 2nd edition.

Ward, B. A., Dutkiewicz, S., and Follows, M. J. (2014). Modelling spatial and temporal patterns in size-structured marine plankton communities: top-down and bottom-up controls. *Journal of Plankton Research*, 36(1):31.

Watson, J. R., Kendall, B. E., Siegel, D. A., and Mitarai, S. (2012). Changing seascapes, stochastic connectivity, and marine metapopulation dynamics. *The American Naturalist*, 180(1):99–112.

Watson, J. R., Mitarai, S., Siegel, D. A., Caselle, J. E., Dong, C., and McWilliams, J. C. (2010). Realized and potential larval connectivity in the Southern California Bight. *Marine Ecology Progress Series*, 401:31–48.

Watson, J. R., Siegel, D. A., Kendall, B. E., Mitarai, S., Rassweiller, A., and Gaines, S. D. (2011). Identifying critical regions in small-world marine metapopulations. *Proceedings of the National Academy of Sciences*, 108(43):E907–E913.

Winemiller, K. O. and Rose, K. A. (1992). Patterns of life-history diversification in North American fishes: implications for population regulation. *Canadian Journal of Fisheries and Aquatic Sciences*, 49(10):2196–2218.

Zakardjian, B. A., Runge, J. A., Plourde, S., and Gratton, Y. (1999). A biophysical model of the interaction between vertical migration of crustacean zooplankton and circulation in the Lower St. Lawrence Estuary. *Canadian Journal of Fisheries and Aquatic Sciences*, 56(12):2420–2432.

Zamarro, J. (1991). Batch fecundity and spawning frequency of yellowtail flounder (*Limanda ferruginea*) on the Grand Bank. *NAFO Scientific Council Studies*, 15:43–51.